Saurashtra University
Re – Accredited Grade 'B' by NAAC
(CGPA 2.93)

Bhatt, Jayeshkumar A., 2010, *"Decision support system in financial statistics for banking industries"*, thesis PhD, Saurashtra University

http://etheses.saurashtrauniversity.edu/id/eprint/354

Copyright and moral rights for this thesis are retained by the author

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.

This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the Author.

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the Author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

Saurashtra University Theses Service
http://etheses.saurashtrauniversity.edu
repository@sauuni.ernet.in

© The Author

Decision Support System

In Financial Statistics for Banking Industries

A Thesis

Submitted To Saurashtra University, Rajkot

In Fulfilment of the Requirement for the Degree Of

Doctor of Philosophy (Statistics) In the Faculty of Science

Submitted by

Jayeshkumar Amritlal Bhatt

Under the guidance of

Dr. D. K. Ghosh

Department Of Statistics

Saurashtra University, RAJKOT

(Gujarat-India)

March – 2010

Certificate by the Guide

This is to certify that the contents of the thesis entitled ***"DECISION SUPPORT SYSTEM IN FINANCIAL STATISTICS FOR BANKING INDUSTRIES"*** *is the original research work of* ***Mr. Jayeshkumar Amritlal Bhatt*** *carried out under my guidance and supervision.*

I further certify that the work has not been submitted either partially or fully to any other University/ Institute for the award of any degree.

Forwarded through supervisor:

Place: Rajkot

Date: 24-03-2010

Dr. D. K. Ghosh
Department Of Statistics
Saurashtra University
RAJKOT-360 005
(Gujarat-India)

Place: Rajkot

Date: 24-03-2010

Dr. D. K. Ghosh
Professor and Head
Department Of Statistics
Saurashtra University
RAJKOT-360 005
(Gujarat-India)

Candidate's Statement

I hereby declare that the work incorporated in the present thesis is original and has not been submitted to any University/ Institution for the award of the Degree. I further declare that the result presented in the thesis, considerations made there-in, contribute in general the advancement of knowledge in education and particular to – "Decision Support System In Financial Statistics for Banking Industries"

Date: 24-03-2010 ***Jayesh Bhatt***

Acknowledgement

I extend my sincere thanks to Dr. D. K. Ghosh, Professor and Head, Department of Statistics, Saurashtra University, Rajkot, who has permitted me to work under his guidance. I take this opportunity to convey my deepest gratitude to him for his valuable advice, encouragement, constructive criticism, scholarly guidance and whole hearted support.

I am highly obliged by the kind hearted bank officer, who has given me the permission and facilities to collect the data from their bank. I admire the assistance of my friends for their help during the data collection, which is finally used for my research work.

I have no words to express my sincere thanks to the authorities and colleagues of my college, i.e., Gurukul Mahila Arts and Commerce College, Porbandar. I can also not forget all the current as well as passed students who are always praying god for the completion of my work as per my ambition and desire. I am grateful to all the well wishers for their encouragement and again thanks them all with due regards.

I must express my gratitude to my parents, wife Suchitra, my lovely loving daughters Heli and Saumya without their enthusiastic and silent support, this work could not have been completed.

Above all, I offer my heartiest regards to god for giving me patience and strength to complete this work.

-Jayesh Bhatt

Paper Publication/ Presentation

1. *A Paper:* ***"Effect of Monthly and Daily Rest Basis on EMI for Housing Loan"*** *published in Spark International Online e-Journal (Volume 2, issue 3, February- August 2010, biannual issue) at www.sparkejournal.blogspot.com, ISSN-0979-7929-U.*

2. *A paper:* ***"Effect of Rest Basis on Bank Deposits"*** *presented in UGC sponsored National Conference on 13th March, 2010 organized by Department of Statistics, Saurashtra University, Rajkot (Gujarat-India).*

Contents

List of figures/ outputs

List of tables/ outputs

Chapter

1

Introduction

1. INTRODUCTION

Computers become more prevalent in everyday life, computer aided office practice is fast becoming the avenue of choice for acquiring ease and accuracy in day to day life. This can bring many benefits to human society. We can conclude that in today's scenario computer mediated processing is the most important solution to help people find useful information and access it in a low-cost and universal manner. This will improve human facilities with the proper decision making for the organization as well as people. Our aim in this thesis is therefore to cover different aspects of the banking like loans, deposits and ATM.

1.1 ABOUT BANK AND ITS FUNCTIONS

We can say Banking is the business of providing financial services to consumers and businesses. The basic services of a bank are checking accounts, which can be used like money to make payments and purchase goods and services; savings accounts and time deposits that can be used to save money for future use; loans that consumers and businesses can use to purchase goods and services; and basic cash management services such as check cashing and foreign currency exchange. Four types of banks specialize in offering these basic banking services: commercial banks, savings and loan associations, savings banks, and credit unions. A broader definition of a bank is any financial institution that receives, collects, transfers, pays, exchanges, lends, invests, or safeguards money for its customers. Banking services serve two primary purposes. First, by supplying customers with the basic mediums-of-exchange (cash, checking accounts, and credit cards), banks play a key role in the way goods and services are purchased. Without these familiar methods of payment, goods could only be exchanged by barter (trading one good for another), which is extremely time-consuming and inefficient. Second, by accepting money deposits from savers and then

lending the money to borrowers, banks encourage the flow of money to productive use and investments. This in turn allows the economy to grow. Without this flow, savings would sit idle in someone's safe or pocket, money would not be available to borrow, people would not be able to purchase cars or houses, and businesses would not be able to build the new factories for which the economy needs to produce more goods and grow. Enabling the flow of money from savers to investors is called financial intermediation, and it is extremely important to a free market economy.

1.1.1 BANKING IN INDIA

Modern banking in India is said to be developed during the British era. In the first half of the 19th century, the British East India Company established three banks – the Bank of Bengal in 1809, the Bank of Bombay in 1840 and the Bank of Madras in 1843. But in the course of time these three banks were amalgamated to a new bank called Imperial Bank and later it was taken over by the State Bank of India in 1955. Allahabad Bank was the first fully Indian owned bank. The Reserve Bank of India was established in 1935 followed by other banks like Punjab National Bank, Bank of India, Canara Bank and Indian Bank.

In 1969, 14 major banks were nationalized and in 1980, 6 major private sector banks were taken over by the government. Today, commercial banking system in India is divided into following categories. The Reserve Bank of India is the central Bank that is fully owned by the Government. It is governed by a central board (headed by a Governor) appointed by the Central Government. It issues guidelines for the functioning of all banks operating within the country.

- **Public Sector Banks**

  State Bank of India and its associate banks called the State Bank Group, Other nationalized banks and Regional rural banks mainly sponsored by public sector banks

- **Private Sector Banks**

  Old generation private banks, new generation private banks, foreign banks operating in India, Scheduled co-operative banks and Non-scheduled banks

- **Co-operative Sector**

  The co-operative sector is very much useful for rural people. The co-operative banking sector is divided into state co-operative banks, central co-operative banks and primary agriculture credit societies.

- **Development Banks/Financial Institutions**

  IFCI, IDBI, ICICI, IIBI, SCICI Ltd., NABARD, Export-Import Bank of India, National Housing Bank, Small Industries Development Bank of India and North Eastern Development Finance Corporation.

1.1.2 FUNCTIONING OF A BANK

Functioning of a Bank is among the more complicated of corporate operations. Since Banking involves dealing directly with money, governments in most countries regulate this sector rather stringently. In India, the regulation traditionally has been very strict. This section, which is also intended for banking professional, attempts to give an overview of the functions in as simple manner as possible.

Banking Regulation Act of India, 1949 defines Banking as "accepting, for the purpose of lending or investment of deposits of money from the public, repayable on demand or otherwise and withdrawable by cheques, draft, order or otherwise."

Deriving from this definition and viewed solely from the point of view of the customers, personal banking, Banks essentially perform the following functions:

1.1.2.1 ACCEPTING DEPOSITS FROM PUBLIC/ OTHERS

1.1.2.1.1 TERM DEPOSIT

Now one can earn a higher income on the surplus funds by investing those with bank. Bank provides security, trust and competitive rate of interest. It is having flexibility in period of term deposit from 15 days to 10 years (generally) and affordable low minimum deposit amount.

1.1.2.1.2 RECURRING DEPOSIT

One can create fund for future planning by investing/ depositing monthly basis. Whatever may be the financial goals, through Recurring Deposit Scheme, one can

save a little amount every month so that at the time of need one will have sufficient funds to achieve financial goals. Recurring Deposit provides the element of compulsion to save at high rates of interest applicable to Term Deposits along with liquidity to access that savings any time. So it is the deposit to set aside a small amount every month and earn at compounded (quarterly) rates of interest.

- Flexibility in period of deposit with maturity ranging from 12 months to 120 months.
- Low minimum monthly deposit amount.

1.1.2.2 LENDING MONEY TO PUBLIC (LOANS)

Lending money is one of the two major activities of any Bank. In a way, the Bank acts as an intermediary between the people who have the money to lend and those who have the need for money to carry out business transactions. Landing money to public is in form of various types of loans like Housing loan, Personal loan, Education loan, Car loan, Loan against mortgage of property etc. Some of the few loans are given below:

1.1.2.2.1 HOUSING LOAN

This can be utilised for purchase/ construction of house/ flat, purchase of a plot of land for construction of house, extension/ repair/ renovation/ alteration of an existing house/ flat, purchase of furnishings and consumer durables as a part of the project cost and takeover of an existing loan from other banks/ housing finance companies.

Most of the friends or relatives I know here in surrounding area has either already bought a house or are planning to buy one. The biggest incentives are perhaps the easy availability of home loans at interest rates far lower than that available to the previous generation and the tax-breaks one gets here in India on the principal and the interest paid on home loans. Our generation is also not averse to taking on a debt unlike the previous generation. In addition, many of us feel that it is better to take the plunge now and enjoy the comforts of your own house than to diligently save all the required money for years and then buy a house, only to find out that the dream was realised a bit too late in your life.

1.1.2.2.2 REDUCING BALANCE LOAN

Suppose you took a Rs 1 lakh loan today at a rate of interest of 10 per cent for five years. You are to pay back Rs 20,000 of the principal and Rs 10,000 (10 per cent of the loan) every year. So you pay back Rs 30,000 every year. Over five years you pay back Rs 1.5 lakh. But notice, that the loan kept reducing over the five years as you paid back Rs 20,000 each year, yet you went on paying interest for five years, as if you had kept the Rs 1 lakh for the entire term.

What if you paid an interest only on the amount you owed each year and not the entire one lakh?

The first year you would pay Rs 10,000 as interest, the next year you would pay Rs 8,000 on a reduced principal of Rs 80,000 and so on, till the last year, you pay only Rs 2,000 as interest. Now you would have paid back Rs 1.3 lakh instead of Rs 1.5 lakh as in the earlier case.

The first case is a situation of a loan that charges interest at a flat rate and the second case is when the interest is calculated on a 'reducing balance' or only on the amount of loan left to pay and not the entire loan amount. A flat 10 per cent is nearly equal to a reducing balance at 6 per cent per annum.

You can get a range of options in reducing balance loans. You get annual, quarterly, monthly, weekly and now daily rests. A 'rest' is jargon to indicate when the bank will recalculate the EMI based on the amount of loan paid back. Suppose you have a loan with an annual 'rest' then, though you pay a monthly instalment, your benefit kicks in only at year end. Meaning the bank gets free interest for 11 months. A monthly 'rest' will recognise the reduction in the loan amount on a monthly basis and a daily 'rest' will do it each day.

1.1.2.2.3 CALCULATING EMIS

When one takes a loan, a natural question that comes to mind is how much the EMI (Equated Monthly Instalment) would be that one has to pay back to the bank every month. Finding out the EMI for different tenures of the home loan allows one to select the best tenure based on one's current and projected income and expenses and possibly other factors.

What I find is that most friends would rather use a bank's EMI calculator rather than calculating it themselves. Some brave souls ask around for the formula to calculate EMIs and then try to write a programme that calculates the EMIs for them. I find it particularly appalling that very few bother to figure it out for themselves, especially since it involves elementary algebra that most of us have surely learnt in high school. It either reflects the creeping sloth and sloppiness in our generation or the rote learning and regurgitating of formulae that our system of education seems to encourage. In any case, I attempt to show how simple it really is to figure out how to calculate EMIs on your own and the effect of monthly rest basis and daily rest basis on it using C++ programming.

For the sake of simplicity, assume that the loan is offered on a "monthly rest" basis. That is, the bank calculates the interest at the end of every month on the amount you still owe to the bank at the beginning of the month, adds it to the amount you already owe and then deducts your EMI from this to calculate the total amount you still owe to the bank at the beginning of the next month. Most of all banks offer loans on a "daily rest" basis, where the outstanding amount and the interest is recalculated every day, but you still pay back on a monthly basis.

Formula of EMI for "monthly rest basis"

Suppose you take on a loan for **P** Rupees, the tenure of the loan is **n** months (for example, n=240 for a 20-year loan), the *monthly rate* of interest is **r** (usually calculated by dividing the annual rate of interest quoted by the bank by 12, the number of months in a year, and dividing that by 100 as the rate is usually quoted as a percentage) and **EMI** Rupees is the EMI you have to pay every month.

Let us use P(i) to denote the amount you still owe to the bank at the end of the i^{th} month. At the very beginning of the tenure, i=0 and P(0)=P, the principal amount you took on as a loan.

At the end of the first month, you owe the bank the original amount P, the interest accrued at the end of the month r×P and you pay back EMI. In other words:

$P(1) = P + r \times P - EMI$ or to rewrite it slightly differently $P(1) = P \times (1 + r) - EMI$

Similarly, at the end of the second month the amount you still owe to the bank is:

$P(2) = P(1) \times (1 + r) - EMI$ or substituting the value of P(1) we calculated earlier:

$$P(2) = (P \times (1 + r) - EMI) \times (1 + r) - EMI$$

and once again expanding it and rewriting it slightly differently:

$$P(2) = P \times (1 + r)^2 - EMI \times ((1 + r) + 1) \quad (1.1)$$

The term "x^y" denotes "x raised to the power y" or "x multiplied by itself y times". To make this look slightly simpler, we substitute "(1 + r)" in equation-(1) by "t" and now it looks like this:

$P(2) = P \times t^2 - EMI \times (1 + t)$

Continuing in this fashion and calculating P(3), P(4) etc. we quickly see that P(i) is given by:

$P(i) = P \times t^i - EMI \times (1 + t + t^2 + ... + t^{i-1})$

At the end of n months (that is, at the end of the tenure of the loan), the total amount you owe to the bank should have become zero. In other words, $P_n=0$. This implies that:

$P(n) = P \times t^n - EMI \times (1 + t + t^2 + ... + t^{n-1}) = 0$, which means that:

$$P \times t^n = EMI \times (1 + t + t^2 + ... + t^{n-1}) \quad (1.2)$$

We can simplify this further by noticing that we have a geometric series of n terms here with a common ratio of t and a scale factor of 1.

The sum of such a series is given by "$(t^n - 1)/(t - 1)$", which we substitute in the above equation-(2) to yield:

$P \times t^n = EMI \times (t^n - 1)/(t - 1)$, which can be rewritten as:

$$EMI = P \times t^n \times (t - 1)/(t^n - 1) \quad (1.3)$$

Equation (3) can again be rewritten by substituting the value of t back as "(1 + r)" as:

$EMI = P \times r \times (1 + r)^n/((1 + r)^n - 1)$ (1.4)

and this is the formula for calculating your EMI.

1.1.2.2.4 EDUCATION LOAN

A term loan granted to Indian Nationals for pursuing higher education in India or abroad where admission has been secured. There will be Comparatively higher interest rates than Housing loan.

Eligible Courses

All courses having employment prospects are eligible, such as graduation courses/ post graduation courses/ professional courses and other courses approved by UGC/ government/ AICTE etc.

Expenses considered for loan

Fees payable to college/school/hostel, Examination/Library/Laboratory fees, Purchase of Books/Equipment/Instruments/Uniforms, Caution Deposit/Building Fund/ Refundable Deposit (maximum 10% tuition fees for the entire course), Travel Expenses/Passage money for studies abroad, Purchase of computers considered necessary for completion of course and cost of a two-wheeler up to Rs. 50,000/- are expenses considered for loan.

1.1.2.2.5 PERSONAL LOAN

Do you want funds readily available to you whenever you desire or need, be it a sudden vacation that you plan with your family or urgent funds required for medical treatment? Personal Loan is the answer to your questions. The loan will be granted for any legitimate purpose whatsoever (e.g. expenses for domestic or foreign travel, medical treatment of self or a family member, meeting any financial liability, such as marriage of son/daughter, defraying educational expenses of wards, meeting margins for purchase of assets etc.) It is having comparatively higher interest rates. Repayment terms generally up to 4-5 years.

1.1.2.3 TRANSFERRING MONEY FROM ONE PLACE TO ANOTHER (REMITTANCES)

Apart from accepting deposits and lending money, Banks also carry out, on behalf of their customers the act of transfer of money - both domestic and foreign.- from one place to another. This activity is known as "remittance business". Banks issue Demand Drafts, Banker's Cheques, Money Orders etc. for transferring the money. Banks also have the facility of quick transfer of money also know as Telegraphic Transfer or Tele Cash Orders.

1.1.2.4 ACTING AS TRUSTEES

Banks also act as trustees for various purposes. For example, whenever a company wishes to issue secured debentures, it has to appoint a financial intermediary as trustee who takes charge of the security for the debenture and looks after the interests of the debenture holders. Such entity necessarily have to have expertise in financial matters and also be of sufficient standing in the market/society to generate confidence in the minds of potential subscribers to the debenture.

1.1.2.5 KEEPING VALUABLES IN SAFE CUSTODY

Bankers are in the business of providing security to the money and valuables of the general public. While security of money is taken care of through offering various type of deposit schemes, security of valuables is provided through making secured space available to general public for keeping these valuables. These spaces are available in the shape of LOCKERS. The latter are small compartments with dual locking facility built into strong cupboards. These are stored in the Bank's Strong Room and are fully secure. Lockers can neither be opened by the hirer or the Bank individually. Both must come together and use their respective keys to open the locker.

1.1.2.6 GOVERNMENT BUSINESS

Earlier Government business used to be exclusively carried out by Governement Treasuries where all type of transactions took place. However, now Banks act on behalf of the Government to accept its tax and non tax receipts. Most of the Government disbursements like pension payments and tax refunds also take place through banks. While the Banks carry out this business for a fee to be paid by the Government,

1.1.2.7 AUTOMATIC TELLER MACHINE-ATM

1.1.2.7.1 ABOUT ATM

With the improved technology, banks provide us another banking method by using the Automatic Teller Machine (ATM).

In 1939, Luther George Simjian patented an early and not-so-successful prototype of an ATM. Some experts have the opinion that James Goodfellow of Scotland holds the earliest patent date of 1966 for a modern ATM, and John D White (also of Docutel) in

the US is often credited with inventing the first free-standing ATM design. In 1967, John Shepherd-Barron invented and installed an ATM in a Barclays Bank in London.

FIGURE 1.1 ATM

Don Wetzel invented an American made ATM in 1968. However, it wasn't until the mid to late 1980s that ATMs became part of mainstream banking.

The Automatic teller machine is good because it helps solve problems for people who don't have time to stand in line waiting inside the bank, or who don't have patience of getting help from the tellers who are performing a slow job at the drive-through window. In order for a customer to make a deposit, you don't need to go inside the bank, but you can still deposit it at any time at the ATMs because it has unlimited hours of operations and there will be a limit of cash that will be available for you to withdraw for the next hours after you deposit. It is also good for people who are traveling around the world, they don't need to carry their cash in their hands, or find their bank locations in order to withdraw cash. It also give you access to your savings account. The ATM machine always provides the sign of card logo's that it accepts before you can insert your card. If you are a non bank customer of a particular that ATM's bank machine, there is a specific fee amount that the bank will charge you in order for you to proceed for further request. For several years, ATM banking services have made it easy for customers to bank without worries. As technology keep on improving, there will be more and more ways that banks will attract people to do business with them. Bankers would still need to be careful with scam, and need to protect their identifications. This banking system will improve even more in years to come.

1.1.2.7.2 TRANSACTIONS USING AN ATM CARD

You can conduct the common transactions like deposits, withdrawals, balance inquiries, mini statements, change your PIN etc. at ATM locations (some transaction types are not available at all ATM locations):

1.1.2.7.3 LIMITATIONS OF ATMS

- Customer can not decide the denominations of notes. Customer can get the notes in predefined denominations. i.e., if amount is more than Rs. 500 then customer will get 5 or less notes of Rs. 100 and remaining notes of Rs. 500.
- Upper limit of withdraw the money per day is fixed by the bank (ranging Rs. 15000 to 40000), so the customer can not utilise the available balance fully.
- Customer can not get the money if the ATM is not loaded with the money.
- Customer can not get the money if the machine is faulty or power supply or link is down.
- Cracking the PIN and card duplication will harm the customer.

1.2 TIME SERIES ANALYIS

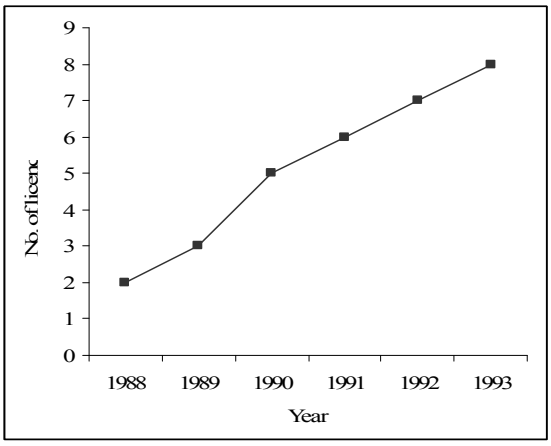
A time series is a set of measurements on a variable taken over some period of time. The time frame of the recorded data may be an hour, a day, a week , a month, a year of a fixed number of years, depending upon the type of event the data refer to. For example, hourly temperature of a particular city, weekly earnings of workers in an industrial town, monthly prices of wheat. Most forecasting techniques involve the use of time series data. In a simple analysis, a time series is assumed to contain the following components.

1.2.1 COMPONENTS

1.2.1.1 THE TREND (SECULAR TREND)

Trend is the long-term tendency of the time series to move in an upward or downward direction. It can be defined as, “A consistent long term change in the average level of the forecast variable per unit of time”. Consider the following figure showing the data on the colour TV licences issued over the period of time.

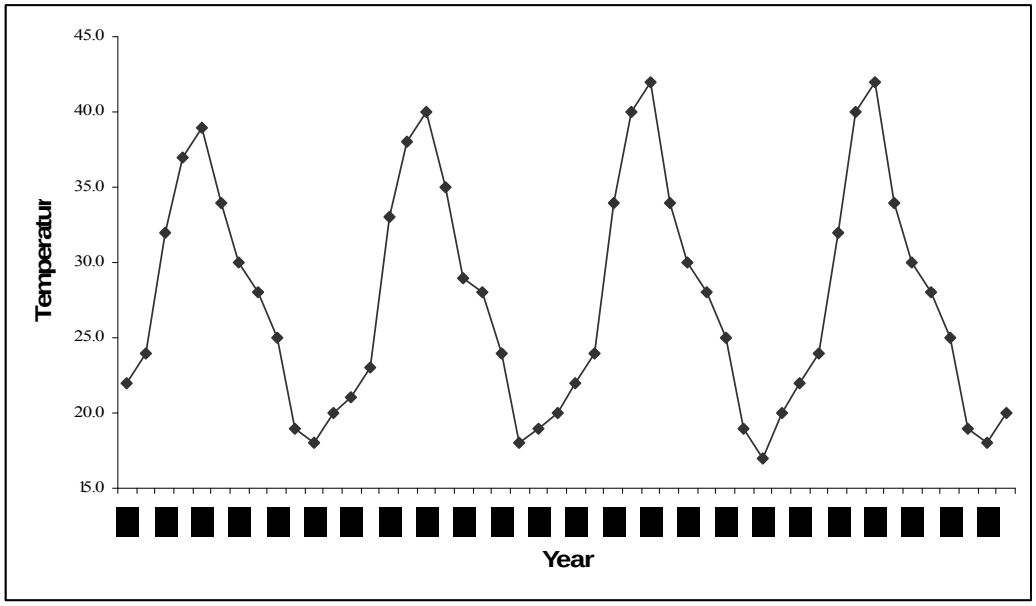
Year	No. of licences
1988	2
1989	3
1990	5
1991	6
1992	7
1993	8



The above data shows a clear upward trend.

1.2.1.2 SEASONAL COMPONENT

Seasonal variations are caused by the seasonal influence of spring, summer, autumn, winter on business and economic activities. Seasonality may also refer to a season in the context of a specific variable on which the data is recorded. For example, in a touring business, season may mean the months of June, July and October, November when number of touring people increase considerable. Thus seasonality may refer to repetition of pattern by months, by weeks, or even by days of the weeks. Consider the following figure showing mean temperature in 'Town A' over the period of time.



Note that the high peaks of 2006 and 2007 reflect the good summer weather of these two years.

1.2.1.3 THE CYCLICAL COMPONENT

Cyclical variations, which are also generally termed as business cycles, are the periodic movements in the time series around the trend line. These are the upswings and downswings in the time series that are observable over extended periods of time. Neither the magnitude nor the frequency of occurrences of these cycles is uniform. For example, A high boom in the stock market in India for a span of four-five years in 1990's caused by the Harshad Mehta factor, can be considered as a cyclical component erecting the activity level of stock market over a period of time.

The cyclical fluctuations are like waves which come and go. At times the cyclical waves prevails for a very long period of time, say 50 years, but these long waves are difficult to be distinguished as they get mixed up with the trend line.

1.2.1.4 THE 'IRREGULAR' OR 'DISTURBANCE' COMPONENT

This is taken to be a random, non systematic component, affecting the time series through such unpredictable events as strikes, breakdown of plant, non-seasonal illness, bad weather etc. These variations either go very deep downward or too high upward to attain peaks abruptly. These do not occur frequently and are less important than the cyclical or seasonal movements.

1.2.2 MODELS OF TIME SERIES ANALYSIS

Analysis of time series involves decomposition of time series into its four components listed above. The objective is to estimate and separate the four types of variations and to bring out the relative impact of each on the overall behaviour of the time series.

There are two models of decomposition of time series:

(i) the additive model (ii) the multiplicative model

1.2.2.1 THE ADDITIVE MODEL

This model is used where it is assumed that the four components are independent of one another. Independence is said to exist when the pattern of occurrences and the magnitude of movements is any particular component are not affected by the other components. Under this assumption, the four components are arithmetically additive in the sense that magnitudes of the time series are the sum of the separate influences of its four components. Thus, if it is taken to represent the magnitude of the time series data at time period t, then Yt can be expressed as:

$$Y_t = T_t + C_t + S_t + R_t$$

Where T_t represents Trend variations, C_t represents Cyclical variations, S_t represents Seasonal variations and R_t represents Random variations

1.2.2.2 THE MULTIPLICATIVE MODEL

This model is used where it is assumed that the forces giving rise to the four types of variations are independent, so that overall pattern of variations in the time series is the combined result of the interaction of all the forces operating on the time series. According to this model, time series are the product of its four components, that is

$$Y_t = T_t \times C_t \times S_t \times R_t$$

As regards the choice between the two models, it is generally the multiplicative model which is used more frequently. The reason being that most business and economic time series data are the result of interaction of a large number of forces which, individually, cannot be treated responsible for generating any particular type of variations. Since the forces responsible for one type of variations are also responsible for the other types of variations, it is the multiplicative model which is ideally suited for the purpose of decomposition of a time series.

1.2.3 MEASUREMENT OF SECULAR TREND

The trend is a concept defining long-term. It does not include short-run oscillations but indicates the steady movements of the variable over a long period of time. But when we record a time series data on a variable, it includes short-run oscillations also. Thus, for measuring the trend values, we first smoothen the data to remove short-term oscillations. For such smoothing any of the following methods may be used:

1. Freehand curve method
2. Method of semi-averages
3. Method of moving averages
4. Method of least squares

1.2.3.1 FREEHAND CURVE METHOD

First we draw a curve after plotting the data of a given time series. It will be irregular as it would include short-run oscillations. We may observe the up and down movement of the curve and smooth out the irregularities by drawing a freehand curve or line along with the curve previously drawn. This freehand curve would eliminate

the short-run oscillations and show the long-period general tendency of the data. This is exactly what is meant by trend.

Note this method has serious disadvantage that different persons may draw the freehand line at different positions and with different slopes. There is, therefore, the danger of different conclusions being draw by different persons.

1.2.3.2 METHOD OF SEMI-AVERAGES

Under this method the whole series is divided into two equal halves and the averages for each half are calculated. If the data is for even number of years, it can easily be divided into two halves. But if it is for odd number of years, we leave the middle year of the time series and two halves constitute the periods on each side of the middle year.

The average for a half is taken to be representative of the value corresponding to the mid point of the time interval of that half. We, thus, get two points. These two points are plotted on a graph and are joined by straight line which provides us the required trend line.

This method is a more objective approach than the free-hand method. But this method is not completely free from criticism. Because we used the mean of each half of the series, any extreme item will greatly affect the points. Such a trend may not be an accurate picture of the growth element in the series. In addition, values so obtained are not precise enough for predictive purpose or for the trend elimination.

1.2.3.3 METHOD OF MOVING AVERAGES

A moving average is an average of fixed number of items in a series which moves through the series by dropping the top item of the previous averaged group and adding the next item below in each successive average. Thus a moving average is computed by adding all the values for a certain number of successive periods and then dividing the sum which is obtained by the number of periods included. This average is considered as the trend value for the unit of time falling at the centre of the period used in the calculation of the average. To compute three yearly moving average, for instance, the values of 1^{st}, 2^{nd} and 3^{rd} years are added up, averaged and the quotient is placed against the 2^{nd} year; then values of the 2^{nd}, 3^{rd} and 4^{th} years are added up, averaged and the average is placed against the 3^{rd} year; and so on.

The moving averages series would yield a smooth curve, provided the time interval chosen for computing the moving average is appropriate. Moving averages may be calculated for odd number of years like 3 years moving average, 5 years moving average, 7 years moving average and for even number of years like 4 years moving average, 6 years moving average and so on. In the case of odd numbers moving average, we place the average against the central year, but when period is even, we follow the 'method of centering'.

The method of moving averages is simple and free from personal bias of the estimator but the choice of the period of moving average needs a great care. If an inappropriate period is selected, a true picture of the trend cannot be obtained. Moreover it involves use of arithmetic average which gets affected by extreme values of the data.

1.2.3.4 METHOD OF LEAST SQUARES

The method of least squares may be used either to fit a linear trend or a non-linear trend. Under this method, a mathematical relationship is established between the time factor X and the variable Y, A straight line of trend is obtained by using the equation for a straight line, which is $Yt = a + bX$.

Where the value of 'a' is merely the Y-intercept or the height of the line above origin. That is, when X=0, Y=a. The other constant 'b' represents the slope of the trend line. When b is positive, the slope is upwards, and when b is negative, the slope is downward.

This line is termed as the line of best fit because it is so fitted that the total distance of deviations of the given data from the line is minimum. The total of deviation is calculated by squaring the difference in trend value and actual value of a variable. Thus, the term "least squares" is attached to this method.

To estimate the value of the constants in the above equation, we start with normal equations. The following are the two normal equations for fitting a straight line by the method of least squares:

$$\sum(y) = na + b\sum(x)$$

$$\sum(xy) = a\sum(x) + b\sum(x)^2$$

1.2.4 MEASUREMENT OF SEASONAL VARIATIONS

The effects of trend cycles and irregular fluctuations must be eliminated from the original time series data to obtain an estimate of seasonal variation. Once they are eliminated, we get measures of seasonal variations in the behaviour of any variable. These measures are called seasonal indexes. By using such indexes, we can deseasonalise the time series. Such deseasonalised data are known as seasonally adjusted data.

There are four methods of constructing seasonal index:

1. Method of Simple Averages
2. Ratio-to-Trend Method
3. Moving Average Method
4. Link Relatives Method

1.2.4.1 METHOD OF SIMPLE AVERAGES

This is the simplest method of obtaining a seasonal index. The following steps are required to calculate index:

(i) Arrange the unadjusted data by months and weeks (say, quarters)

(ii) Find the totals of all quarters for January, February, etc.

(iii) Divide each total by the number of months for which data are given and get quarterly average.

(iv) Obtain general average of quarterly/weekly averages.

(v) Compute seasonal index as follows:

$$\text{Seasonal Index for 1st quarter} = \frac{\text{Quarterly average for 1st quarter}}{\text{General average}} \times 100$$

This method is simplest of all the methods of estimating the seasonal variations. But it is not a scientific method. The computation is based on the assumption that there is no trend component in the given time series. Thus the resultant seasonal index is a combination of trend and seasonal variations.

1.2.4.2 RATIO TO TREND METHOD

This method isolates the seasonal factor after eliminating the trend from tie series. Trend is eliminated by computing the ratios, and random elements disappear when the ratios are averaged. The steps involved in computation of index are:

1. Determine trend value by the method of least squares.
2. Divide the original data quarter by quarter, by the corresponding trend values and express them as percentages.
3. Average the different values for a quarter.
4. Adjust all these averages.

It may be noted here that step (2) eliminates the trend and the values so obtained include cyclical and irregular variations. Step (3) frees the values from cyclical and irregular variations.

1.2.5 MEASUREMENT OF CYCLICAL VARIATIONS: Residual Method

Since the original data are represented by $T_t \times S_t \times C_t \times R_t$ we can obtain the cyclical irregular movements by eliminating trend and seasonal variation, that is

$$\frac{T_t \times S_t \times C_t \times R_t}{T_t \times S_t} = C_t \times R_t$$

The remaining data $(C \times R)$ are usually smoothed out to obtain the cyclical variations, which are called the cyclical relatives or percentages. This is done by the method of moving averages, and the irregular component is removed in the process of averaging. As a result, what is left in the series is merely the cyclical movements. This explains why the procedure is commonly referred to as the residual method. Steps are:

1. Multiply the trend values by seasonal index.
2. The original figures of the time series are divided by resultant products, resulting in the combination of the cyclical and irregular movements.
3. For smoothing out irregular movement from $C_t \times R_t$ data, obtain moving average from the $C_t \times R_t$ data.
4. C is left as a residue.

1.2.6 MEASUREMENT OF IRREGULAR VARIATIONS

The irregular component in a time series represents the residue of fluctuations after trend, cyclical and seasonal movements have been accounted for. It may be represented as follows:

$$\frac{Y}{T_t \times C_t \times S_t} = \frac{T_t \times C_t \times S_t \times R_t}{T_t \times C_t \times S_t} = R_t$$

It is simple to measure the irregular fluctuations, once the trend, seasonal and cyclical variations have been measured.

1.3 WEIBULL DISTRIBUTION

This distribution is found to be of great importance in the fields of reliability and life testing. Perhaps this is the most widely used life time distribution. As early as 1928 researchers in the theory of extreme value knew this distribution to which the name weibull is given. This distribution had been derived earlier by Fisher and Tippett (1928). This is also known as Fisher-Tippett type III distribution. The Weibull distribution arises in a natural way from the exponential distribution if we assume that β^{th} power of failure time has exponential distribution. Weibull (1951) showed this distribution is useful in decreasing the 'wear out' or fatigue failures. The work may be classified under the various categories as follows:

1.3.1 DEFINITION

The life time random variable x is said to have Weibull distribution if its probability density function is given by

$$f(x,\beta,\theta) = \frac{\beta}{\theta} x^{\beta-1} \exp\left(-\frac{x^{\beta}}{\theta}\right) \text{ where } x>0,\ \beta>0,\ \theta>0 \qquad (1.3.1)$$

β is the shape parameter and θ is the scale parameter, which is known as two-parameter Weibull distribution.

1.3.2 MAXIMUM LIKELIHOOD ESTIMATION

Cohen (1965) suggested the following method by which we can obtain Maximum Likelihood Estimates of β and θ by means of iterative procedure.

Let $x_1, x_2, x_3, \ldots, x_n$ be a random sample of size n from w(0, β, θ) distribution. The Likelihood function of this sample is

$$L(x_1, x_2, x_3, \ldots, x_n; \beta\ \theta) = \prod_{i=1}^{n} \frac{\beta}{\theta} x_i^{\beta-1} \exp\left\{-\frac{x_i^{\beta}}{\theta}\right\}$$

Taking logarithm we get

$$\text{Ln L}= \text{n ln } \beta - \text{n ln } \theta + (\beta\text{-}1) \sum_{i=1}^{n} \ln x_i - \frac{\sum_{i=1}^{n} x_i^{\beta}}{\theta} \tag{1.3.2}$$

Differentiating log likelihood with respect to the parameters β and θ and equating to zero we obtain

$$\frac{\partial \ln L}{\partial \beta} = \frac{n}{\beta} + \sum_{i=1}^{n} \ln x_i - \frac{\sum_{i=1}^{n} x_i^{\beta} \ln x_i}{\theta} = 0 \tag{1.3.3}$$

$$\frac{\partial \ln L}{\partial \theta} = -\frac{n}{\beta} + \frac{\sum_{i=1}^{n} x^{\beta}}{\theta^2} = 0 \tag{1.3.4}$$

$$\frac{\partial^2 \ln L}{\partial \beta^2} = \frac{-n}{\beta^2} - \frac{n\left(\sum_{i=1}^{n} x_i^{\beta} (\ln x_i)^2\right)}{\left(\sum_{i=1}^{n} x_i^{\beta}\right)} \tag{1.3.5}$$

$$\frac{\partial^2 \ln L}{\partial \theta^2} = \frac{n}{\theta^2} - \frac{2}{\theta^3} \sum_{i=1}^{n} \left(x_i^{\beta}\right) \tag{1.3.6}$$

$$\frac{\partial^2 \ln L}{\partial \beta \partial \theta} = \frac{\partial^2 \ln L}{\partial \theta \partial \beta} = \frac{1}{\theta^2} \sum_{i=1}^{n} \left(x_i^{\beta} \ln x_i\right) \tag{1.3.7}$$

$$I(\theta) = E\begin{pmatrix} -\frac{\partial^2 \ln L}{\partial \beta^2} & -\frac{\partial^2 \ln L}{\partial \beta \partial \theta} \\ -\frac{\partial^2 \ln L}{\partial \theta \partial \beta} & -\frac{\partial^2 \ln L}{\partial \theta^2} \end{pmatrix}_{(\hat{\beta},\hat{\theta})} \tag{1.3.8}$$

$$\Sigma = E(I(\theta))^{-1}$$

$$K = E\left(-\frac{\partial^2 \ln L}{\partial \hat{\beta}^2}\right) E\left(-\frac{\partial^2 \ln L}{\partial \hat{\theta}^2}\right) - E\left(-\frac{\partial^2 \ln L}{\partial \hat{\beta} \partial \hat{\theta}}\right) E\left(-\frac{\partial^2 \ln L}{\partial \hat{\theta} \partial \hat{\beta}}\right)$$

$$\Sigma = \frac{1}{K}\begin{pmatrix} E\left(-\frac{\partial^2 \ln L}{\partial \theta^2}\right) & E\left(-\frac{\partial^2 \ln L}{\partial \beta \partial \theta}\right) \\ E\left(-\frac{\partial^2 \ln L}{\partial \theta \partial \beta}\right) & E\left(-\frac{\partial^2 \ln L}{\partial \beta^2}\right) \end{pmatrix}_{(\hat{\beta},\hat{\theta})}$$

1.3.3 GRAPHICAL METHOD OF GETTING INITIAL SOLUTION

We have,

$$\bar{F}(x) = e^{\frac{-x^{\beta}}{\theta}}$$

$$\ln \bar{F}(x) = \frac{-x^{\beta}}{\theta}$$

$$-\ln \bar{F}(x) = \frac{x^{\beta}}{\theta}$$

$$\ln\left(-\ln \bar{F}(x)\right) = -\ln\theta + \beta \ln x$$

$$y = \mu + \beta t$$

Where, t=ln x and μ = -ln θ

For given data $x_1, x_2, x_3, \ldots, x_n$ we obtain $t_i = \ln x_i$, i=1,2,3,…. (1.3.9)

The survival function $\bar{F}(x)$ is calculated using Kaplan-meier estimator as

$$\hat{\bar{F}}\left(x_{(i)}\right) = \frac{n-i+1}{n+1}$$

Define $s_i = \ln(-\ln \bar{F}(x_{(i)}))$

$$= \ln(\ln(\frac{n+1}{n-i+1})) \quad (1.3.10)$$

By taking (s_i, t_i), i=1, 2, 3, …, n we fit a straight line.

Then estimator of slope (which is same as shape parameter)

$$\hat{\beta} = \frac{\sum (s_i - \bar{s})(t_i - \bar{t})}{\sum (t_i - \bar{t})^2} \quad (1.3.11)$$

And

$$\hat{\theta} = \begin{cases} \dfrac{\sum x_i^{\beta}}{n}, (uncensored) \\ \dfrac{\sum x_i^{\hat{\beta}} + (n-r)x_{(r)}^{\hat{\beta}}}{r}, (censored) \end{cases} \quad (1.3.12)$$

1.3.3.1 MEAN AND VARIANCE OF WEIBULL DISTRUBUTION

The mean and variance of the two parameters Weibull distribution are given by:

$$\mu_1' = \theta^{\frac{1}{\beta}}\Gamma\left(\frac{\beta+1}{\beta}\right) \tag{1.3.13}$$

$$\mu_2 = \theta^{\frac{2}{\beta}}\left[\Gamma\left(\frac{\beta+2}{\beta}\right) - \Gamma^2\left(\frac{\beta+1}{\beta}\right)\right] \tag{1.3.14}$$

The Coefficient of Variation (C.V.) is independent of θ and is given by

$$C.V. = \frac{\sqrt{\mu_2}}{\mu_1'} = \frac{\left[\Gamma\left(\frac{\beta+2}{\beta}\right) - \Gamma^2\left(\frac{\beta+1}{\beta}\right)\right]^{1/2}}{\Gamma\left(\frac{\beta+1}{\beta}\right)} \tag{1.3.15}$$

1.3.4 ESTIMATION OF PARAMETERS USING N-R METHOD

Newton's method (also known as the Newton–Raphson method), named after Isaac Newton and Joseph Raphson, is perhaps the best known method for finding successively better approximations to the zeroes (or roots) of a real-valued function. Newton's method can often converge remarkably quickly, especially if the iteration begins "sufficiently near" the desired root with the graphical method.

Given a function $f(x)$ and its derivative $f'(x)$, we begin with a first guess x_0 .

A better approximation x_1 is

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Similarly we can have $x_2, x_3, \ldots, x_n$ as under:

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)},\ x_3 = x_2 - \frac{f(x_2)}{f'(x_2)},\ \ldots,\ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Let us understand this method using one simple example of finding square root of a number:

If we wish to find the square root of 5, this is equivalent to find the solution to

$$x^2 = 5,$$

So, the function used in N-R method is then

$$f(x) = x^2 - 5$$

With derivative $f'(x) = 2x$

With an initial guess of $x_0 = 2$, the sequence of iteration to find the solution given by N-R method is as under:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 2 - \frac{2^2 - 5}{2(2)} = 2.25$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} = 2.25 - \frac{2.25^2 - 5}{2(2.25)} = 2.2361111111$$

$$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} = 2.2361111111 - \frac{2.2361111111^2 - 5}{2(2.2361111111)} = 2.2360679779$$

$$x_4 = x_3 - \frac{f(x_3)}{f'(x_3)} = 2.2360679779 - \frac{2.2360679779^2 - 5}{2(2.2360679779)} = 2.2360679774$$

$$x_5 = x_4 - \frac{f(x_4)}{f'(x_4)} = 2.2360679774 - \frac{2.2360679774^2 - 5}{2(2.2360679774)} = 2.2360679774$$

It is quite remarkable that the results stabilize for ten decimal places after only 5 iterations!

1.4 ABOUT C/ C++ PROGRAMMING LANGUAGE

In computing, **C** is a general-purpose, block structured, procedural, imperative computer programming language developed in 1972 by Dennis Ritchie at the Bell Telephone Laboratories for use with the Unix operating system.

Although C was designed for implementing system software, it is also widely used for developing application software.

It is widely used on a great many different software platforms and computer architectures, and several popular compilers exist. C has greatly influenced many

other popular programming languages, most notably C++, which originally began as an extension to C.

1.4.1 EARLY DEVELOPMENTS

The initial development of C occurred at AT&T Bell Labs between 1969 and 1973; according to Ritchie, the most creative period occurred in 1972. It was named "C" because many of its features were derived from an earlier language called "B", which according to Ken Thompson was a stripped-down version of the BCPL programming language.

The origin of C is closely tied to the development of the UNIX operating system, originally implemented in assembly language on a PDP-7 by Ritchie and Thompson, incorporating several ideas from colleagues. Eventually they decided to port the operating system to a PDP-11. B's lack of functionality to take advantage of some of the PDP-11's features, notably byte addressability, led to the development of an early version of the C programming language.

The original PDP-11 version of the Unix system was developed in assembly language. By 1973, with the addition of `struct` types, the C language had become powerful enough that most of the UNIX kernel was rewritten in C. This was one of the first operating system kernels implemented in a language other than assembly.

Short history of the C family of languages is as under:

1972 - The precursor to C, the language B, is developed at Bell Labs. The B language is fast, easy to maintain, and useful for all kinds of development from systems to applications. The entire team that designed the language is immediately fired for behavior unbefitting a telephone company employee, and the project is handed to Dennis Ritchie. He alters the language to be incomprehensible, difficult to maintain, and only useful for systems development. He also designs in a pointer system guaranteed to give every program over 500 lines a pointer into the operating system.

1982 – It is discovered that 97% of all C routine calls are subject to buffer overrun exploits. C programmers begin to realize that initializing a variable to whatever happens to be lying around in memory is not necessarily a good

idea. However, since enforcing sensible variable initialization would break 97% of all C programs in existence, nothing is done about it.

1984 – The number of operating systems has been dramatically increased.

1985 – A variant of C with object oriented capabilities, called C With Classes, is ready to go commercial. However, the name C With Classes is considered too clear and easy for outsiders to understand, so the commercial version is called C++.

1986 – C becomes so popular that industry analysts recommend writing business applications in it. They argue that applications written in C will be portable to many different systems. Many of these industry analysts are suspected of being under the influence of hallucinogens.

1988 – Industry analysts finally run out of LSD. After their hallucinations fade, they notice that business apps written in C take five times longer to produce, and are still not portable. They stop recommending that business apps be written in C, except for a minority that switch to crack cocaine and start recommending business apps be written in C++ because "object orientation will result in code reuse".

By this time, all C compilers have turned into C++ compilers and we can say C is a subset of C++, here we have also used C++ compiler.

1.4.2 DATA TYPES

As a programming language, C is rather like Pascal or Fortran. Values are stored in variables. Programs are structured by defining and calling functions. Program flow is controlled using loops, if statements and function calls. Input and output can be directed to the terminal or to files. Related data can be stored together in arrays or structures. Of the three languages, C allows the most precise control of input and output. C is also rather more terse than Fortran or Pascal. This can result in short efficient programs, where the programmer has made wise use of C's range of powerful operators. It also allows the programmer to produce programs which are impossible to understand.

Care must be taken in using C. Many of the extra facilities which it offers can lead to extra types of programming error. While dealing with large real numbers we should

be very careful for their data type and range. Let us have a brief understanding about data types available in C. Here we will have brief note about data types, Primary data type, Integer Type, Floating Point Types, Void Type, Character Type, Size and Range of Data Types on 16 bit machine. A C language programmer has to tell the system before-hand, the type of numbers or characters he is using in his program. These are data types. There are many data types in C language. A C programmer has to use appropriate data type as per his requirement. Data types available in C language are like:

1.4.2.1 INTEGER TYPES

Integers are whole numbers with a machine dependent range of values. A good programming language supports the programmer by giving a control on a range of numbers and storage space. C has 3 classes of integer storage namely short int, int and long int. All of these data types have signed and unsigned forms. A short int requires half the space than normal integer values. Unsigned numbers are always positive and consume all the bits for the magnitude of the number. The long and unsigned integers are used to declare a longer range of values.

1.4.2.2 FLOATING POINT TYPES

Floating point number represents a real number with 6 digits precision. Floating point numbers are denoted by the keyword float. When the accuracy of the floating point number is insufficient, we can use the double to define the number. The double is same as float but with longer precision. To extend the precision further we can use long double which consumes 80 bits of memory space.

1.4.2.3 VOID TYPE

Using void data type, we can specify the type of a function. It is a good practice to avoid functions that does not return any values to the calling function.

1.4.2.4 CHARACTER TYPE

A single character can be defined as a defined as a character type of data. Characters are usually stored in 8 bits of internal storage. The qualifier signed or unsigned can be explicitly applied to char. While unsigned characters have values between 0 and 255, signed characters have values from –128 to 127.

1.4.3 SIZE AND RANGE OF DATA TYPES ON 16 BIT MACHINE

TABLE 1.1

TYPE	SIZE (Bits)	Range
Char or Signed Char	8	-128 to 127
Unsigned Char	8	0 to 255
Int or Signed int	16	-32768 to 32767
Unsigned int	16	0 to 65535
Short int or Signed short int	8	-128 to 127
Unsigned short int	8	0 to 255
Long int or signed long int	32	-2147483648 to 2147483647
Unsigned long int	32	0 to 4294967295
Float	32	3.4 e-38 to 3.4 e+38
Double	64	1.7e-308 to 1.7e+308
Long Double	80	3.4 e-4932 to 3.4 e+4932

1.4.4 THE MATHEMATICS OF FLOATING POINT ARITHMETIC

A big problem with floating point arithmetic is that it does not follow the standard rules of algebra. One of the primary goals of this note is to describe the limitations of floating point arithmetic so you will understand how to use it properly.

Consider Range of the data type: Normal algebraic rules apply only to infinte precision arithmetic. Consider the simple statement x=x+1, x is an integer. On any modern computer this statement follows the normal rules of algebra as long as overflow does not occur. That is, this statement is valid only for certain values of x (minint <= x < maxint).

Automatic type conversion: Very important characteristic of C is automatic type conversion. One can lose the fractional part by assigning any floating point value to a variable of integer data type.

Integer Arithmetic: If all the operands in arithmetic expression are integers then this will become integer arithmetic and the result will be integer only. Most programmers do not have a problem with this because they are well aware of the fact that integers in a program do not follow the standard algebraic rules. e.g., value of 5/2 is 2, here it is not 2.5.

Internal representation: To represent real numbers, most floating point formats employ scientific notation and use some number of bits to represent a mantissa and a

smaller number of bits to represent an exponent. The end result is that floating point numbers can only represent numbers with a specific number of significant digits. This has a big impact on how floating point arithmetic operations. The mantissa and exponents are both signed values

Greater precision: For the greater precision we should use the proper data type to have more significant digits and to avoid rounding errors. The accuracy loss during a single computation usually isn't enough to worry about unless we are greatly concerned about the accuracy of our computations. However, if we compute a value which is the result of a sequence of floating point operations, the error can accumulate and greatly affect the computation itself. For example, suppose we were to compute certain process for more number of iterations we must have higher precision arithmetic. It will be better to have long double data type instead of float or double in certain case.

The order of evaluation: The order of evaluation can effect the accuracy of the result. By applying normal algebraic transformations, you can arrange a calculation so the multiply and divide operations occur first. For example, suppose you want to compute x*(y+z). Normally you would add y and z together and multiply their sum by x. However, you will get a little more accuracy if you transform x*(y+z) to get x*y+x*z and compute the result by performing the multiplications first.

Overflow or Underflow: Multiplication and division are not without their own problems. When multiplying two very large or very small numbers, it is quite possible for overflow or underflow to occur. The same situation occurs when dividing a small number by a large number or dividing a large number by a small number

Comparison of floating point numbers: Comparing floating pointer numbers is very dangerous. Given the inaccuracies present in any computation (including converting an input string to a floating point value), we should never compare two floating point values to see if they are equal. The test for equality succeeds if and only if all bits (or digits) in the two operands are exactly the same. Since this is not necessarily true after two different floating point computations which should produce the same result, a straight test for equality may not work.

The standard way to test for equality between floating point numbers is to determine how much error (or tolerance) you will allow in a comparison and check to see if one value is within this error range of the other. The straight-forward way to do this is to use a test like the following:

if Value1 >= (Value2-error) and Value1 <= (Value2+error) then ...

Another common way to handle this same comparison is to use a statement of the form: if abs(Value1-Value2) <= error then ...

1.5 OBJECTIVES

The main objective of this investigation is to frame the structure, details of files, tabulation, calculations and charts using Excel, algorithm of problem and write complete program in C++ for the following files as shown in table 1.2.

TABLE 1.2 List of C++ programs

Sr.	File name	Description
1	daytot.cpp	Read the data file "ymdtamt.csv" having collected data of ATM and create a output text file "dailytot.txt" having day wise total number of hits, total amount withdrawn and average amount withdrawn.
2	mmlytot.cpp	To read the data file "ymdtamt.csv" having collected data of ATM and create a output text file "mmlytot.txt" having monthly total number of hits, total amount and average amount and also to generate the output showing month results on screen.
3	qmahit.cpp	To read the data file "mmlytot.txt" and create a text file "qmahit.txt" having quarterly moving average of hits. Also generates well formatted table showing quarterly moving average of hits on the screen.
4	qmaamt.cpp	To read the data file "mmlytot.txt" and create a text file "qmaamt.txt" having quarterly moving average of amount withdrawn. Also generates the well formatted table showing quarterly moving average of amount withdrawn on the screen.
5	irrvartn.cpp	To read the data file "ymdtamt.csv" having collected data of ATM and create output files for hits "irrvar.txt" and "leastsq.txt". Also displays tables for fitting of straight line for trend value, weekly trend values, weekly percentage of hits and cyclical irregularities of hits.

6	irramt.cpp	To read the data file "ymdtamt.csv" and create output files for amount withdrawn "irramt.txt" and "leastamt.txt". Also displays tables for fitting of straight line for trend value, weekly trend values, weekly percentage of amount withdrawn and cyclical irregularities of amount withdrawn.
7	lsq2nd.cpp	Fitting of second degree curve for hits.
8	lsq2sol.cpp	Solution of a second degree equation of trend for hits.
9	lsq2ndam.cpp	Fitting of second degree curve for amount withdrawn.
10	lsq2sola.cpp	Solution of a second degree equation of trend for amount withdrawn.
11	emimmly.cpp	To generate EMIs for various rate of interests and tenures on the basis of monthly compounding.
12	emidaily.cpp	To generate EMIs for various rate of interests and tenures on the basis of daily compounding.
13	emidiff.cpp	To generate the difference table showing difference between EMIs based on daily compounding and monthly compounding for various rate of interest and tenures.
14	emicheck.cpp	To generate a table to calculate closing balance for the given loan amount, rate and EMI after completion of 5 years (tenure).
15	emichkal.cpp	To generate a table showing summary of closing balances for given loan amount and rate for the EMIs of different tenures.
16	fdqtrly.cpp	To generate a table showing maturity amounts for given deposit amount for various rates and tenures, on quarterly compounded basis.
17	fddaily.cpp	To generate a table showing maturity amounts for given deposit amount for various rates and tenures, on daily compounded basis.
18	fddiff.cpp	To generate a table showing difference of maturity amount for given deposit amount for various rates and tenures, on daily compounded basis and quarterly compounded basis.
19	recurqfl.cpp	To generate a data file showing maturity amounts for various rates and tenures for the recurring of Rs.100 every month, on quarterly compounded basis.
20	recurdfl.cpp	To generate a data file showing maturity amounts for various rates and tenures for the recurring of Rs.100 every month, on daily compounded basis.

21	recudiff.cpp	To generate a data file showing difference of maturity amounts for various rates and tenures for the recurring of Rs.100 every month, on daily compounded basis and quarterly compounded basis.
22	grfile.cpp	To generate a data file for the calculation of initial solution of MLEs using graphical method - Weibull distribution for hits.
23	nrtable1.cpp	To generate iterations of N-R method for the final solution of estimation of parameters - Weibull distribution for hits.
24	gramt.cpp	To generate a data file for the calculation of initial solution of MLEs using graphical method - Weibull distribution for amount withdrawn per day.
25	nramt.cpp	To generate iterations of N-R method for the final solution of estimation of parameters - Weibull distribution for amount withdrawn per day.
26	anovahit.cpp	To generate ANOVA table for hits.
27	anovaamt.cpp	To generate ANOVA table for amount withdrawn.

This will help the general public in finding

1. The EMI for the loan taken from the bank or any other financial institution.
2. The proper tenure for the loan using EMIs.
3. To find the amount calculated on the basis of daily and quarterly compounding basis so that one can find how much the customer is loosing if the bank gives interest on the basis of quarterly compounding instead of daily compounding. Similarly for the amount taken from the bank.

So far such type of investigation for the bank or any financial institution has not been made in past hence we took keen interest in such finding with the help of bank, internet and website. The program developed in C++ in this investigation can be made as software for the future plan.

From this investigation we concluded that how much amount a bank in the ATM can deposit every day month wise or season wise so that customer will not return from the ATM of the corresponding bank without taking required money. Which implies that bank can be kept self in safety side by providing that much money in their ATM based on mode in statistics.

To achieve these objectives, this thesis is structured as follows.

In chapter 1 we have covered general introduction regarding complete thesis. Mainly we have covered following points in this chapter:

- About Bank and Its Functions
- Time Series Analysis
- Weibull distribution
- About C/C++ programming language

Chapter 2 describes the research methodology and data collection. We have collected the ATM data from a bank located at Porbandar for the period of one year. In this chapter we have described various aspects of data collection and handling like structure of the collected data, data entry, data validations, and conversions of original data into required format etc., also shown about the programming language C/C++.

Chapter 3 describes the time series analysis of hits and amount withdrawn from the ATM. Here we have given general introduction of time series analysis. We have also discussed various measurements of secular trends like freehand curve method, method of moving averages, method of least squares and measurement of seasonal variations like ratio to trend method. We have written C++ programs for these methods. We have used Excel to draw various charts for these methods.

As we know that computers become more prevalent in everyday life, computer aided office practice is fast becoming the avenue of choice for acquiring ease and accuracy in day to day life. This will improve human facilities with the proper decision making for the organization as well as people. Banking sector is one of the leading sectors, where such office practice is very useful. The basic services a bank provides are checking accounts, which can be used like money to make payments and purchase goods and services; savings accounts and time deposits that can be used to save money for future use; loans that consumers and businesses can use to purchase goods and services; and basic cash management services such as check cashing and foreign currency exchange. Banks are also called custodians of public money. Lending money is one of the two major activities of any Bank. In a way, the Bank acts as an intermediary between the people who have the money to lend and those who have the need for money to carry out business transactions. Landing money to public is in form

of various types of loans like Housing loan, Personal loan, Education loan, Car loan etc. In chapter 4 we have calculated EMIs for different rates of interest and different tenures on the basis of monthly and daily compounding using C++. We have shown the effect of rest basis by various difference tables.

Now one can earn a higher income on the surplus funds by investing those with bank. The quantum of interest depends upon the rate of interest, term - length of time for which the depositor wishes to keep the money with the Bank and method of compounding. One can calculate maturity amount using different methods of compounding. Here in chapter 5 we have calculated maturity amounts of various bank deposits using method of quarterly compounding and method of daily compounding using programs written in C++. We have shown the effect of rest basis by various difference tables of maturity amounts.

As we have discussed in chapter 2 that we have collected ATM data for 1 year i.e. for the period 01-01-2005 to 31-12-2005. Our main focus of the study is on number of hits per day (number of customers visited to the ATM) and the amount withdrawn per hit. In chapter 6 we have covered the fitting of Weibull distribution on hits and amount withdrawn, we have applied method of maximum likelihood Estimation for the estimation of parameters β and θ. Using C++ programs we have calculated initial solution using graphical method. We have also generated all the iterations of N-R method and found final solution using C++ program. We have also found the final solution using goal seek method of Excel. We have calculated mean and variance also. We have prepared the tables for goodness of fit test and found that hits and amount withdrawn per day of a bank at Porbandar fits to the Weibull distribution.

Chapter

2

Research Methodology and Data Collection

2.1 DATA COLLECTION FROM BANK

In this age of global world, the ATM is good because it helps to solve problems for which people don't have time to stand in line waiting inside the bank or who do not have patience of getting help from the tellers who are performing a slow job at the window. It is also good for people who are travelling different places; they do not need to carry their cash in their hands. It also gives you access to your saving account. Thus ATMs are playing major role in customer services.

ATMs are providing many facilities, but we have focused our study only on withdrawal of amount i.e. Study the number of customers visited per day and amount withdrawn.

The most difficult task in achieving the objectives said was collection of data. The data being confidential, it was quite essential to maintain the secrecy of the data as well as the bank branch. I have visited so many banks, discussed with the concern officers about our objectives and requested for the required data. I found it very difficult to convince them due to their limitations. Ultimately I found one cooperative and academic minded officer, who convinced with our objectives and permitted me to collect the data from the bank.

Initially ATM machines are filled with blank paper roles, the details of each and every transaction will be printed on the paper role, attached to the ATM. Printed paper roles are dumped in the store room and replaced with the blank role 2-3 times in a month or as and when necessary. We have focused only on the withdrawals and found date of transaction, arrival time and amount withdrawal for each transaction i.e. each visit of the customer.

That gentle, cooperative and academic minded officer has given me facility in the branch to write these details.

We have noted down the data for about

365 days or 12 months or 17824 hits/ records/ withdrawals.

Considering our duty hours and bank pick hours, most of the data collection was done after their regular timings only. All the transaction history was printed on a paper role continuously; proper care was required to read each transaction without skipping any transaction or duplication of transaction. It was a very lengthy and time consuming process to write each and every transaction in the note book.

2.2 DATA PREPARATION IN EXCEL WORKSHEET

2.2.1 DATA ENTRY

Here we have selected Microsoft Excel worksheet for the data handling. We have created three columns, each for date, time and amount withdrawn. Obviously the data was ordered date wise and time wise chronologically. There are in total 17824 hits for amount withdrawal so there are 17824 rows (records) and 3 columns in the worksheet.

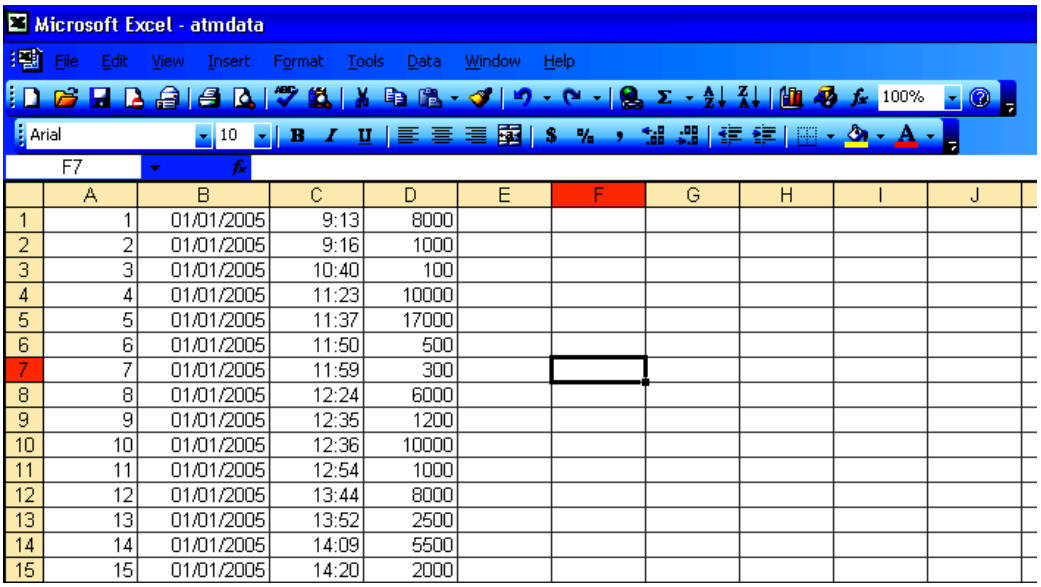
2.2.2 DATA VALIDATION

As we are human being, there is a chance of incorrect entry. We have validated the data, using facilities available in Excel, in following ways:

- Current time must not be less than the previous time, if date is same.
- Amount withdrawn per hit should not exceed the maximum limit. (Rs.40000.00)
- Amount should be positive.

Sample of the initial worksheet of the collected data is as under:

FIGURE 2.1



	A	B	C	D	E	F	G	H	I	J
1	1	01/01/2005	9:13	8000						
2	2	01/01/2005	9:16	1000						
3	3	01/01/2005	10:40	100						
4	4	01/01/2005	11:23	10000						
5	5	01/01/2005	11:37	17000						
6	6	01/01/2005	11:50	500						
7	7	01/01/2005	11:59	300						
8	8	01/01/2005	12:24	6000						
9	9	01/01/2005	12:35	1200						
10	10	01/01/2005	12:36	10000						
11	11	01/01/2005	12:54	1000						
12	12	01/01/2005	13:44	8000						
13	13	01/01/2005	13:52	2500						
14	14	01/01/2005	14:09	5500						
15	15	01/01/2005	14:20	2000						

2.2.3 CONVERSION OF DATE AND TIME IN NUMERIC TYPE

We want some arithmetic process on date and time, so we have converted them into numeric format.

Dates are converted as integer numbers. Considering 01-01-1900 as the first day, 01-01-2005 is 38353rd day; 02-01-2005 is 38354th day and so on.

Considering one day as integer 1, each hour can become 24th part of 1 i.e. 0.041667. For every one hour there will be in increment of 0.041667. Following table 2.1 shows numeric conversion of hour.

TABLE 2.1

Time Format	Number Format
1:00 AM	0.041667
2:00 AM	0.083333
3:00 AM	0.125000
4:00 AM	0.166667
5:00 AM	0.208333
6:00 AM	0.250000
7:00 AM	0.291667
8:00 AM	0.333333
9:00 AM	0.375000
10:00 AM	0.416667
11:00 AM	0.458333
12:00 NOON	0.500000
1:00 PM	0.541667
2:00 PM	0.583333
3:00 PM	0.625000
4:00 PM	0.666667
5:00 PM	0.708333
6:00 PM	0.750000
7:00 PM	0.791667
8:00 PM	0.833333
9:00 PM	0.875000
10:00 PM	0.916667
11:00 PM	0.958333
12:00 AM	1.000000

Each minute is 60^{th} part of an hour i.e. 0.041667/60 = 0.000694

or $24 \times 60 = 1440^{th}$ part of a day, i.e. 1/1440=0.000694.

So after every minute there will be an increment of 0.000694. Following table 2.2 shows numeric conversion of minutes.

TABLE 2.2

Time Format HH:MM	01:01	01:02	01:03	01:04	01:05	01:06	…
Number Format	0.042361	0.043056	0.043750	0.044444	0.045139	0.045833	…

Overall date + time can be converted as a fractional number.

Initially we had 3 columns in the worksheet. We have generated three new columns like year of the date, month of the date and day of the date from the date field.

2.2.4 FORMATTING OF DATE AND TIME IN EXCEL

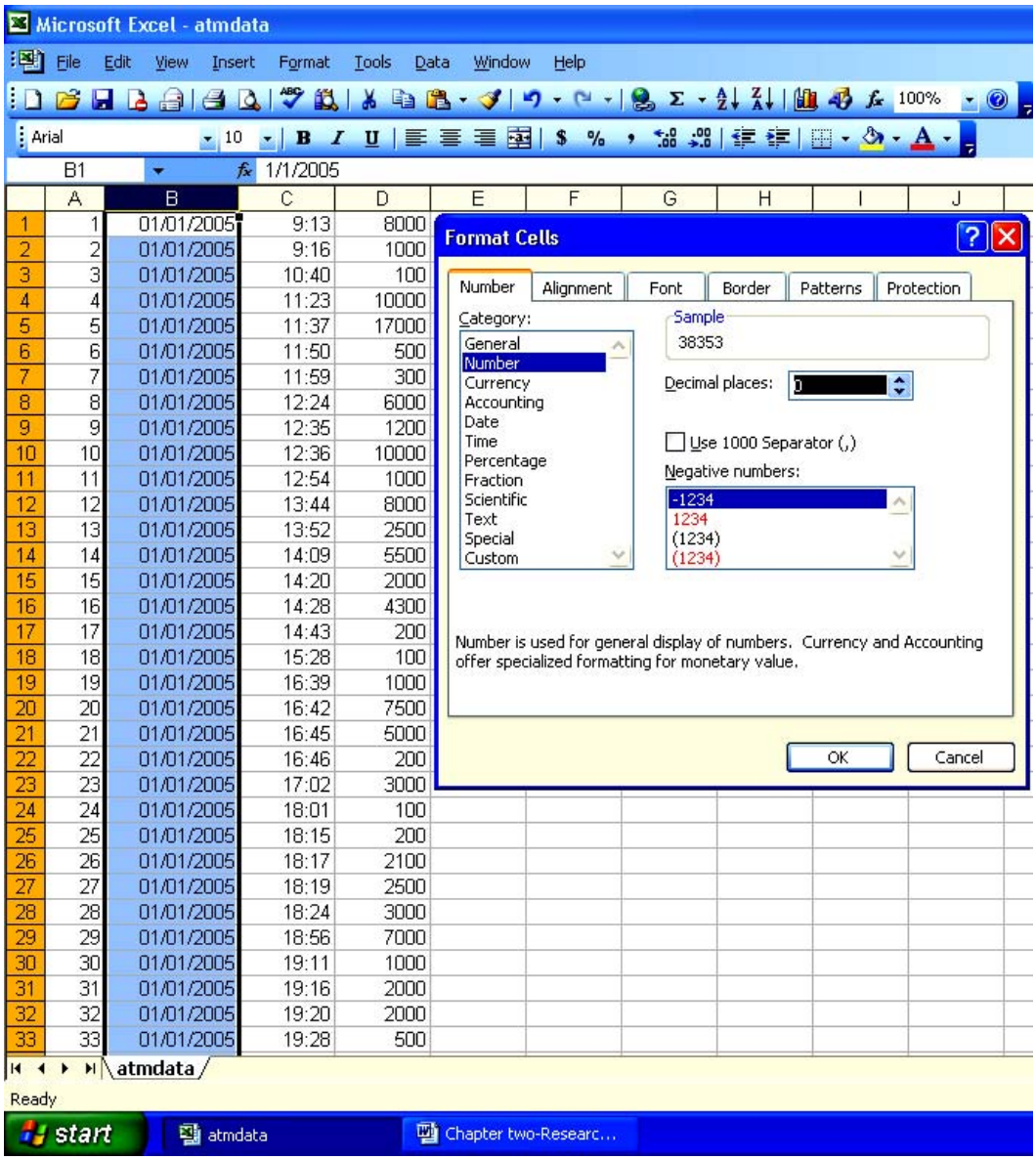
We have entered the date in DD/MM/YYYY format and time in HH:MM format. For the analysis purpose we require these in numeric format. By this means date will be converted as integer serial number and time will be converted as a fractional part of a day. The steps of formatting are as under:

1) Format the date column as numeric type

   a) Select the column of date

   b) Using cells of format menu, format the date as shown in figure 2.2.

FIGURE 2.2

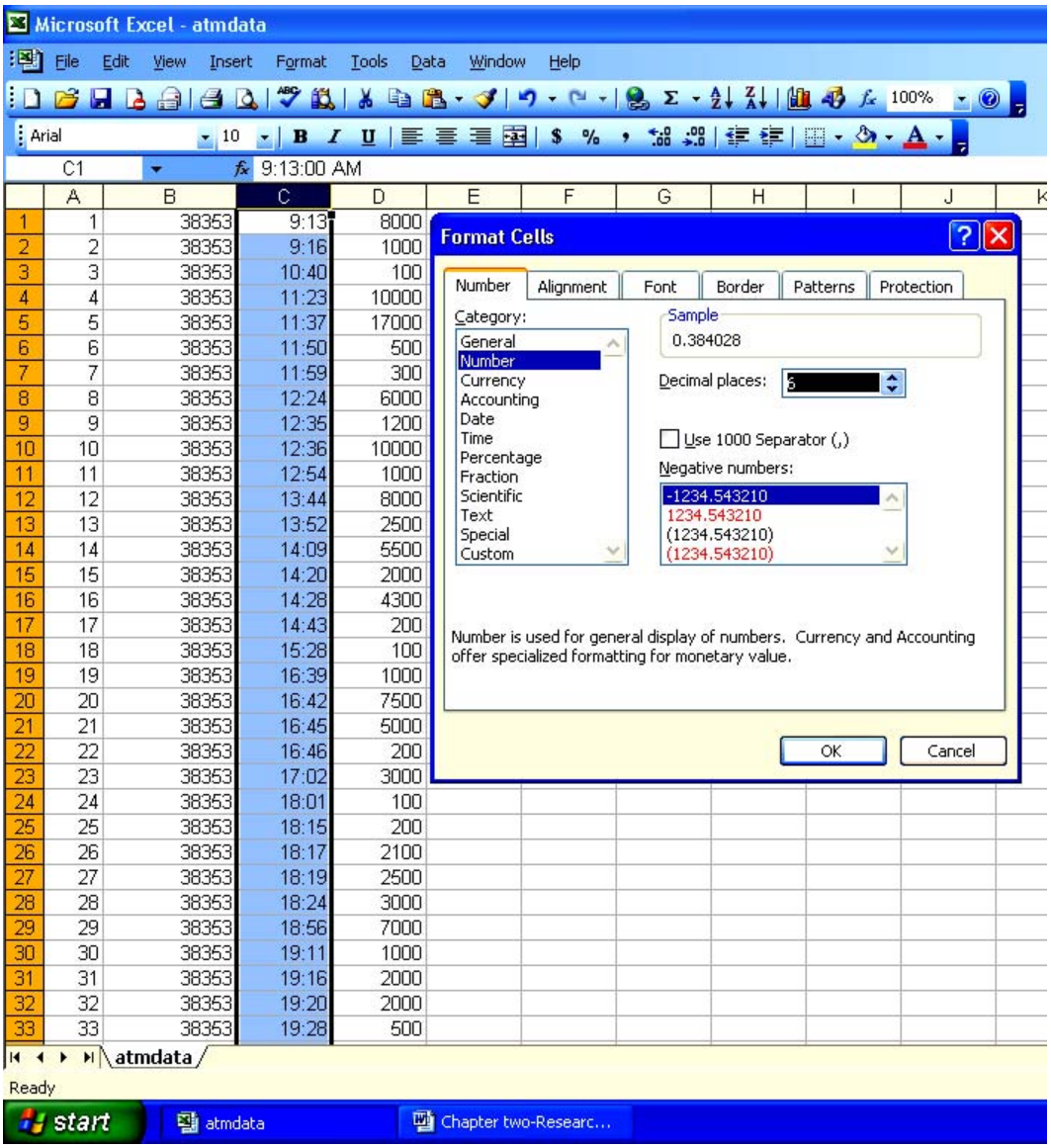


2) Format the time column as numeric type

   a) Select the column of time

   b) Using cells of format menu, format the time as shown in figure 2.3.

FIGURE 2.3



3) Separate year, month and day of the date

a) Insert three columns B,C and D each for year, month and day

b) Get the year of the date by writing the formula =YEAR(E1) in B1

c) Get the month of the date by writing the formula =MONTH(E1) in C1

d) Get the day of the date by writing the formula =DAY(E1) in D1

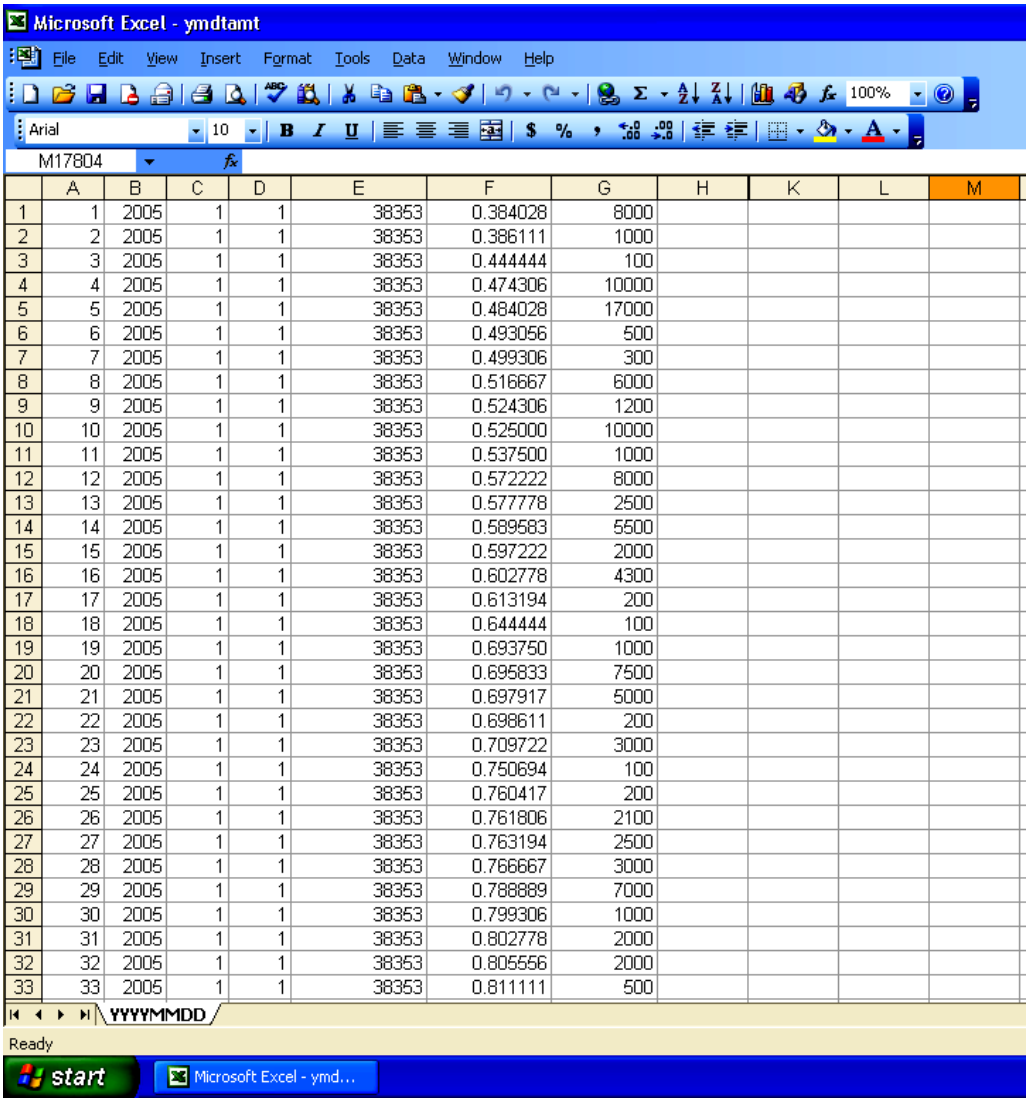
e) Copy B1, C1 and D1 for all the dates

f) Finally we have created the worksheet '*ymdtamt.xls*' file having columns like Serial number, Numeric year of the date i.e. YEAR(E1), Numeric month of the date i.e. MONTH(E1), Numeric day of the date

i.e. DAY(E1), Date as number format, Time as number format and Amount.

Sample of the worksheet after conversion in above mentioned format is as under at figure 2.4:

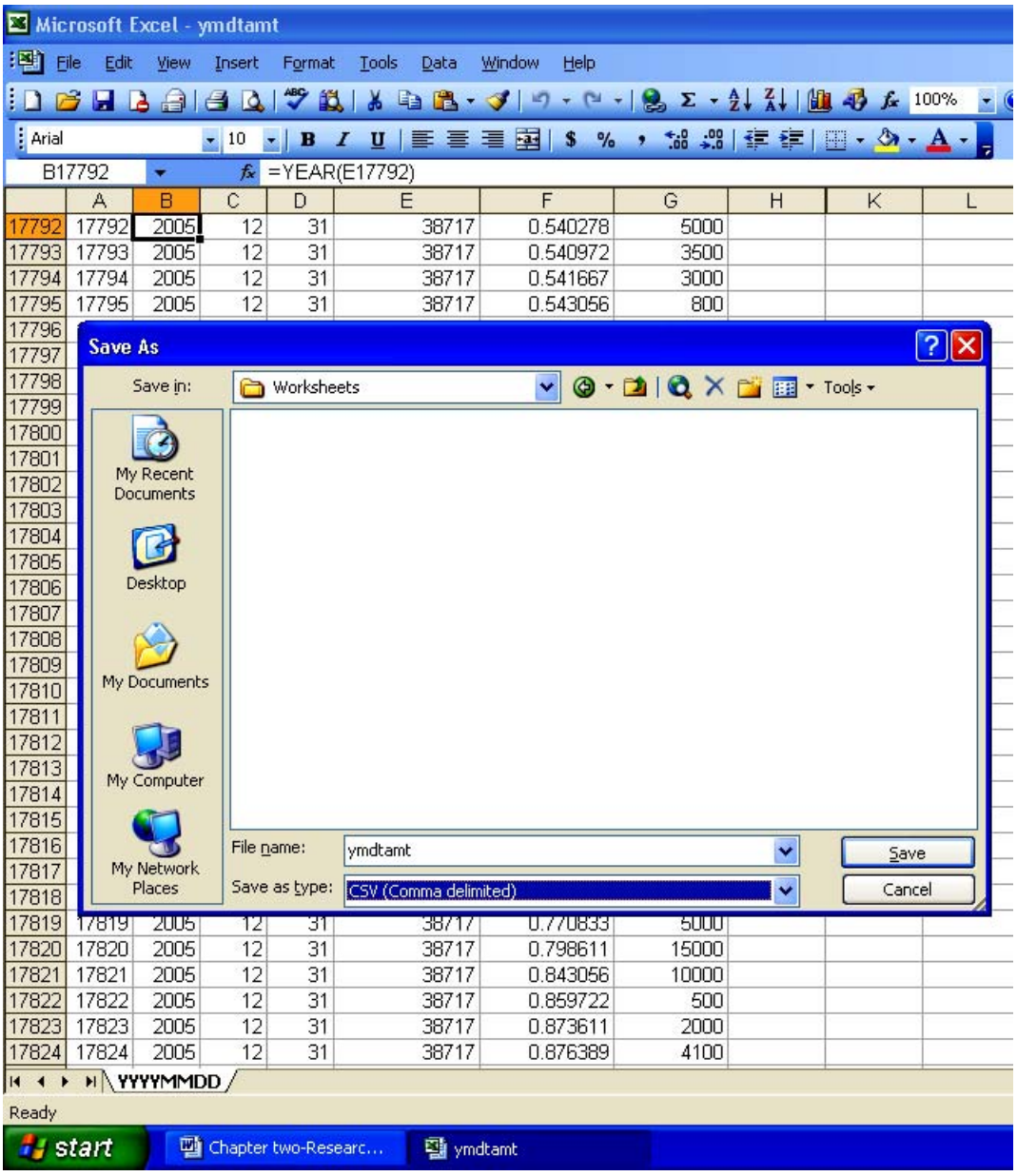
FIGURE 2.4



	A	B	C	D	E	F	G	H	K	L	M
1	1	2005	1	1	38353	0.384028	8000				
2	2	2005	1	1	38353	0.386111	1000				
3	3	2005	1	1	38353	0.444444	100				
4	4	2005	1	1	38353	0.474306	10000				
5	5	2005	1	1	38353	0.484028	17000				
6	6	2005	1	1	38353	0.493056	500				
7	7	2005	1	1	38353	0.499306	300				
8	8	2005	1	1	38353	0.516667	6000				
9	9	2005	1	1	38353	0.524306	1200				
10	10	2005	1	1	38353	0.525000	10000				
11	11	2005	1	1	38353	0.537500	1000				
12	12	2005	1	1	38353	0.572222	8000				
13	13	2005	1	1	38353	0.577778	2500				
14	14	2005	1	1	38353	0.589583	5500				
15	15	2005	1	1	38353	0.597222	2000				
16	16	2005	1	1	38353	0.602778	4300				
17	17	2005	1	1	38353	0.613194	200				
18	18	2005	1	1	38353	0.644444	100				
19	19	2005	1	1	38353	0.693750	1000				
20	20	2005	1	1	38353	0.695833	7500				
21	21	2005	1	1	38353	0.697917	5000				
22	22	2005	1	1	38353	0.698611	200				
23	23	2005	1	1	38353	0.709722	3000				
24	24	2005	1	1	38353	0.750694	100				
25	25	2005	1	1	38353	0.760417	200				
26	26	2005	1	1	38353	0.761806	2100				
27	27	2005	1	1	38353	0.763194	2500				
28	28	2005	1	1	38353	0.766667	3000				
29	29	2005	1	1	38353	0.788889	7000				
30	30	2005	1	1	38353	0.799306	1000				
31	31	2005	1	1	38353	0.802778	2000				
32	32	2005	1	1	38353	0.805556	2000				
33	33	2005	1	1	38353	0.811111	500				

As we have used the C++ programming language, we require the data file accessible through C++. To prepare the data file accessible to C++ we have saved the worksheet in CSV format as under at figure 2.5:

FIGURE 2.5



Now, our data is ready for further processing. We have developed C++ programs to read the data file ymdtamt.csv and generated various data files and results for the different objectives.

Chapter

3

Time Series Analysis of Hits and Amount Withdrawn

3.1 INTRODUCTION

Most business problems contain variables whose values are random over time. Time, either as an independent variable or as an added dimension to the existing variables may help the decision making in business. In decision making problems time may be one of the most important variables. This usually occurs in problems where we wish to estimate the expected value of y or to predict the new value of y at a future point of time after having observed the pattern of the values of y during past and present time periods. Here we have collected ATM data like date and time of visit of the customer and amount withdrawn by him for the period of one year. Analysis of time series is of great significance as it helps in understanding the past behaviour.

3.2 MEASUREMENT OF SECULAR TREND

3.2.1 FREEHAND CURVE METHOD

To draw the charts for day wise and month wise time series data, we have created a data files for the daily totals and monthly totals. For the creation of required data files we have written following programs in C++. The file name of the first program is 'daytot.cpp', whose aim is to read the data file 'ymdtamt.csv' and create a text file 'dailytot.txt' having day wise total number of hits, total amount withdrawn and average amount withdrawn. Let us discuss the structure of files below:

Structure of files

ymdtamt.csv		dailytot.txt	
Name of the field	Data Type	Name of the field	Data Type
Serial number	Integer	Day serial number	Integer
Numeric year of the date	Integer	Day of the date	Integer
Numeric month of the date	Integer	Month of the date	Integer
Numeric day of the date	Integer	Year of the date	Integer

Date as number format	Long integer	Total number of hits	Integer
Time as number format	Double	Total amount of withdrawal	Double
Amount	Double	Average amount of withdrawal	Double

Here we cite a sample data for 15 days in the month of January 2005. The data are shown in Table 3.1.

TABLE 3.1

Day serial number	Day of the date	Month of the date	Year of the date	Total number of hits	Total amount of withdrawal	Average amount of withdrawal
1	1	1	2005	41	129000	3146.34
2	2	1	2005	31	58500	1887.10
3	3	1	2005	36	115900	3219.44
4	4	1	2005	33	194700	5900.00
5	5	1	2005	30	127100	4236.67
6	6	1	2005	31	143100	4616.13
7	7	1	2005	27	98800	3659.26
8	8	1	2005	30	171200	5706.67
9	9	1	2005	23	75800	3295.65
10	10	1	2005	27	105200	3896.30
11	11	1	2005	34	132000	3882.35
12	12	1	2005	43	150600	3502.33
13	13	1	2005	41	184300	4495.12
14	14	1	2005	21	32700	1557.14
15	15	1	2005	28	109700	3917.86

However we have collected the actual data for the period 01-01-2005 to 31-12-2005.

Initially for this program ‘daytot.cpp’, let the algorithm is given below:

1. Open an input data file1 “ymdtamt.csv” to read in “r” mode
2. Open an output data file2 “dailytot.txt” to write in “w” mode
3. Check for the end of the file in file1, if end of file then display appropriate message and terminate the process.
4. If not end of file then set the counter 1 to serial number of day, dayno.
5. Read one by one record while not end of file and repeat following steps.
    a. Set counters for total hits and total amount to zero.
    b. Read one by one record and repeat following steps for the day of date is not changed.

i. Add amount to total amount.

ii. Increment the counter of total hits by one.

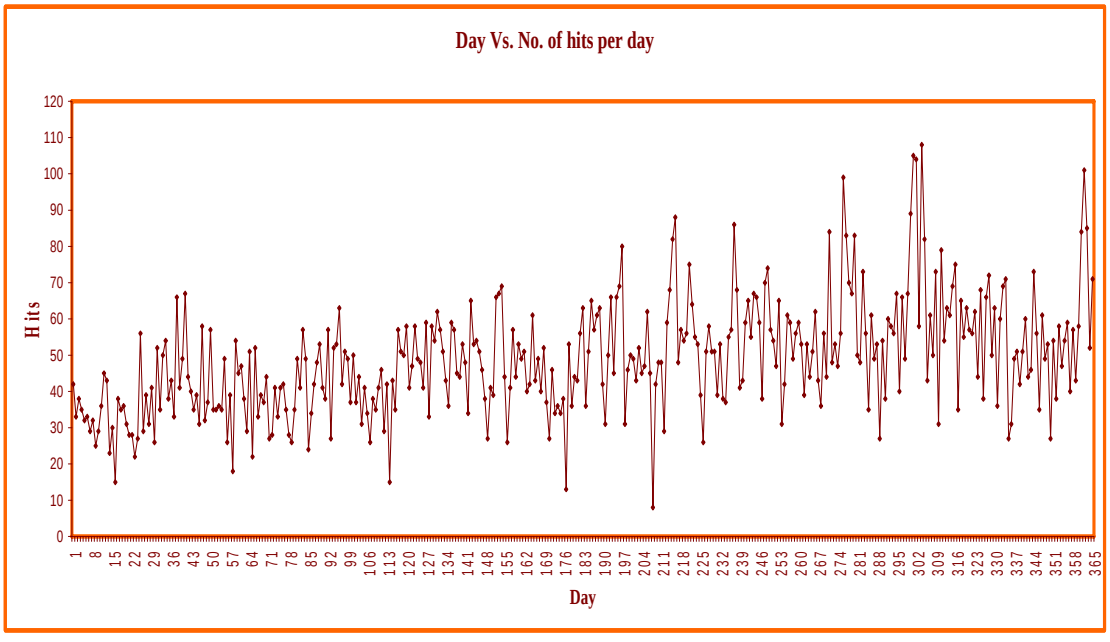
c. Calculate average daily amount of withdrawal. Divide total amount by total number of hits and store the result to average daily amount.

d. Write serial number of the day, day of date, month of date, year of date, total number of hits per day, total amount withdrawn per day and average amount per hit per day to data file2.

6. Close the data file1 and data file2 and terminate the process.

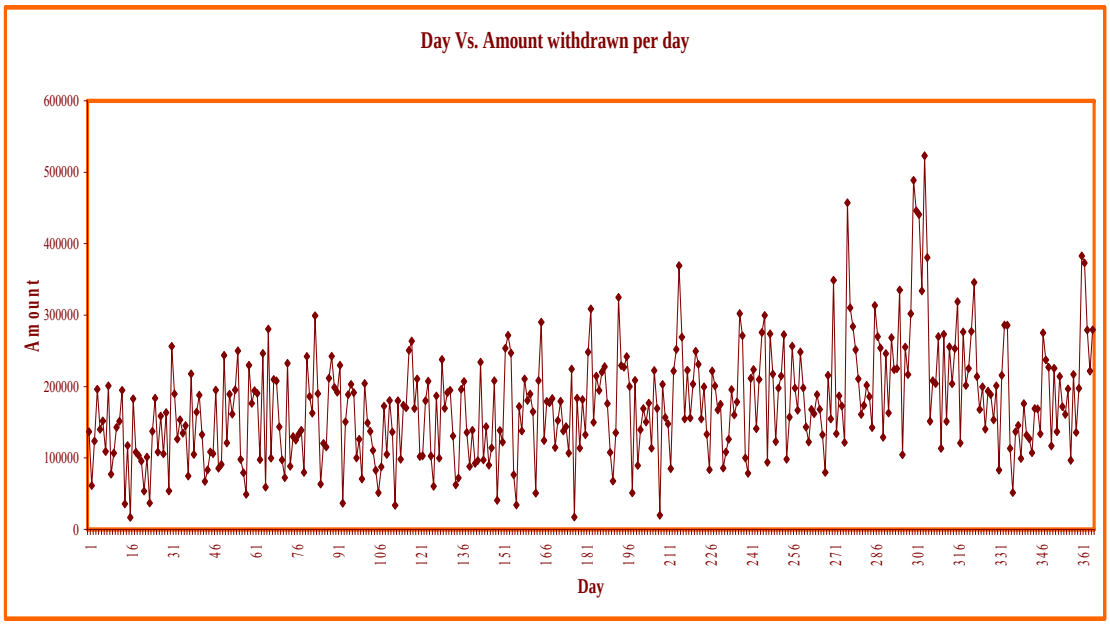
As a result of above program 'daytot.cpp', the file named "dailytot.txt" was created. This file contains Day serial number, Day of the date, Month of the date, Total number of hits, Total amount of withdrawal and Average amount of withdrawal. To study the pattern of hits and amount withdrawn we have drawn charts showing day vs. number of hits per day, Day vs amount withdrawn per day and Day vs. average amount withdrawn per day per hit. To draw charts from the stored data in the file, we have opened this text file in Excel named 'dailytot.xls' and stored as Excel format. We have generated following charts in Excel.

CHART 3.1



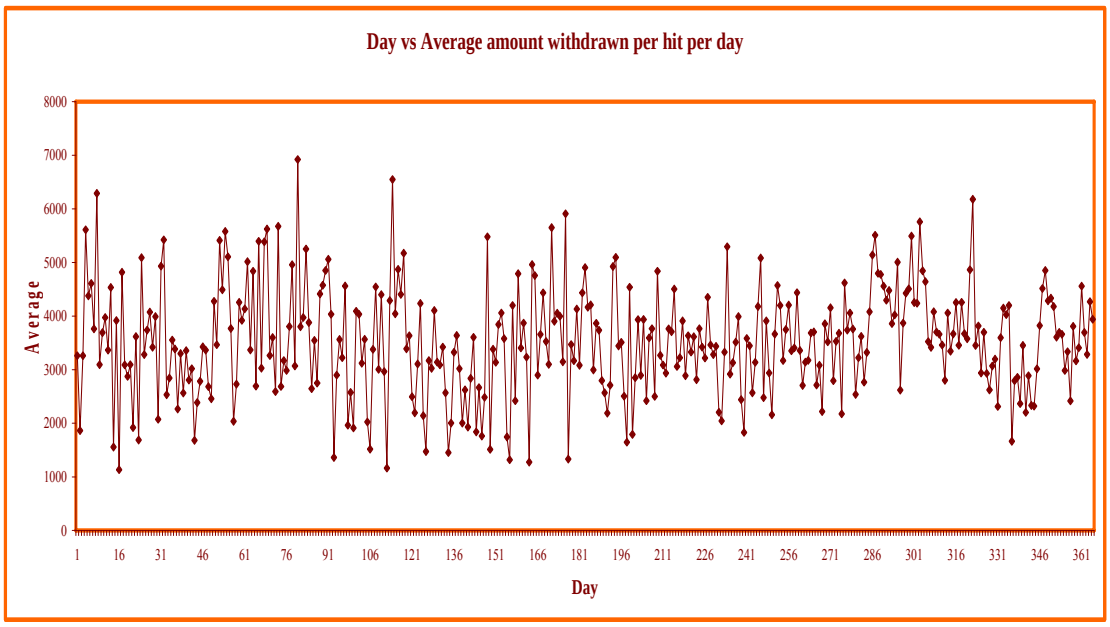
From chart 3.1 it is clear that maximum number of hits is 108 on 304th day i.e. 31-10-2005 and minimum number of hits is 8 on 208th day i.e. 27-07-2005.

CHART 3.2



From chart 3.2 it is clear that maximum amount withdrawn is Rs.523100 on 304th day i.e. 31-10-2005 and minimum amount withdrawn is Rs.17000 on 16th day i.e. 16-01-2005.

CHART 3.3



From chart 3.3 it is clear that maximum average amount withdrawn is Rs.6920 on 80th day i.e. 21-03-2005 and minimum average amount withdrawn is Rs.1133.33 on 16th day i.e. 16-01-2005.

To read the data file "ymdtamt.csv" and create a text file "mmlytot.txt" having monthly total number of hits, total amount and average amount and also to generate the output showing month results on screen, we have written a C++ program having file name 'mmlytot.cpp'.

Later we discuss the structure of files below:

ymdtamt.csv		mmlytot.txt	
Name of the field	Data Type	Name of the field	Data Type
Serial number	Integer	Month number	Integer
Numeric year of the date	Integer	Total number of hits	Integer
Numeric month of the date	Integer	Total amount of withdrawal	Double
Numeric day of the date	Integer	Average amount of withdrawal	Double
Date as number format	Long integer		
Time as number format	Double		
Amount	Double		

Algorithm for the program 'mmlytot.cpp' is given below:

1. Open a data file1 "ymdtamt.csv" to read in "r" mode
2. Open a data file2 "mmlytot.txt" to write in "w" mode
3. Check for the end of the file in file1, if end of file then display appropriate message and terminate the process.
4. If not end of file then read one by one record while not end of file and repeat following steps.
    a. Set counters for total hits and total amount to zero.
    b. Read one by one record and repeat following steps for the month is not changed.
        i. Add amount to total amount.
        ii. Increment the counter of total hits by one.
    c. Calculate average monthly amount. Divide total amount by total number of hits and store the result to average monthly amount.
    d. Write the month number, total number of hits, total amount and average amount to data file2.
    e. Print the output in formatted way on the screen.
5. Close the data file1 and data file2.
6. Terminate the process.

Using program 'mmlytot.cpp' we have displayed following output on the screen shown at figure 3.1:

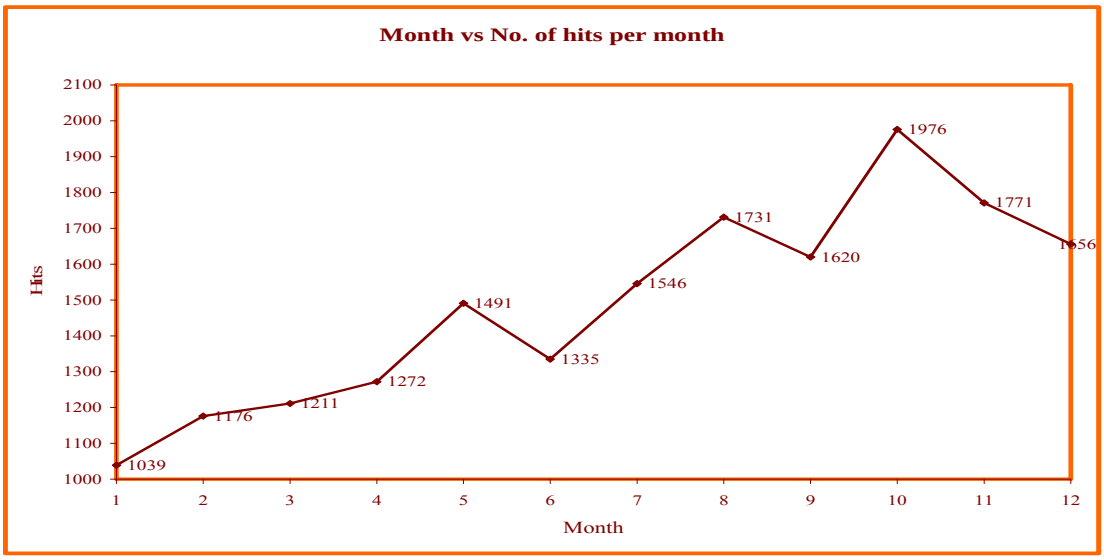
FIGURE 3.1

Month Number	Total no. of hits	Total Amount(Rs.)	Average Amount(Rs.)
1	1039	3816300.00	3673.05
2	1176	3988900.00	3391.92
3	1211	5038400.00	4160.53
4	1272	4460400.00	3506.60
5	1491	4248000.00	2849.09
6	1335	4842900.00	3627.64
7	1546	5379950.00	3479.92
8	1731	5851070.00	3380.17
9	1620	5709700.00	3524.51
10	1976	8215700.00	4157.74
11	1771	6562430.00	3705.49
12	1656	5709100.00	3447.52

Grand total of hits=17824

Opened the above created resulting file in Excel named 'mmlytot.xls' and drawn following charts.

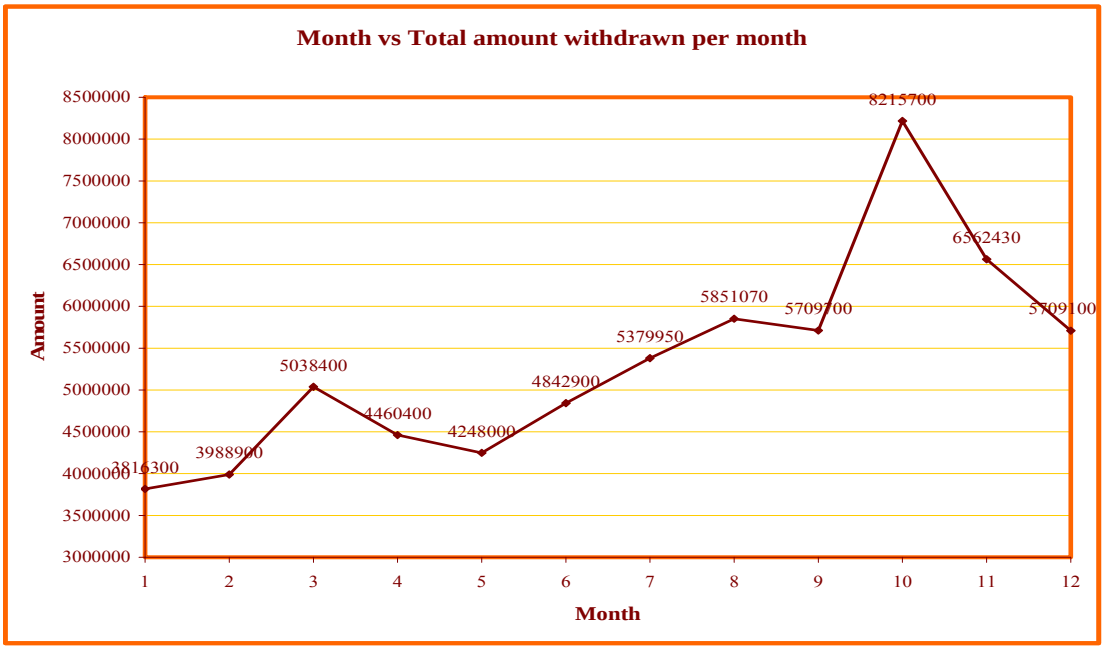
CHART 3.4



CONCLUSION

From chart 3.4 it is clear that number of hits increases every month. It is falling down in certain month like June and September, may be due to specific reason. During the months like May, August and October it is increased which may be due to vacation period and festivals like Janmashtami and Navaratri/Diwali.

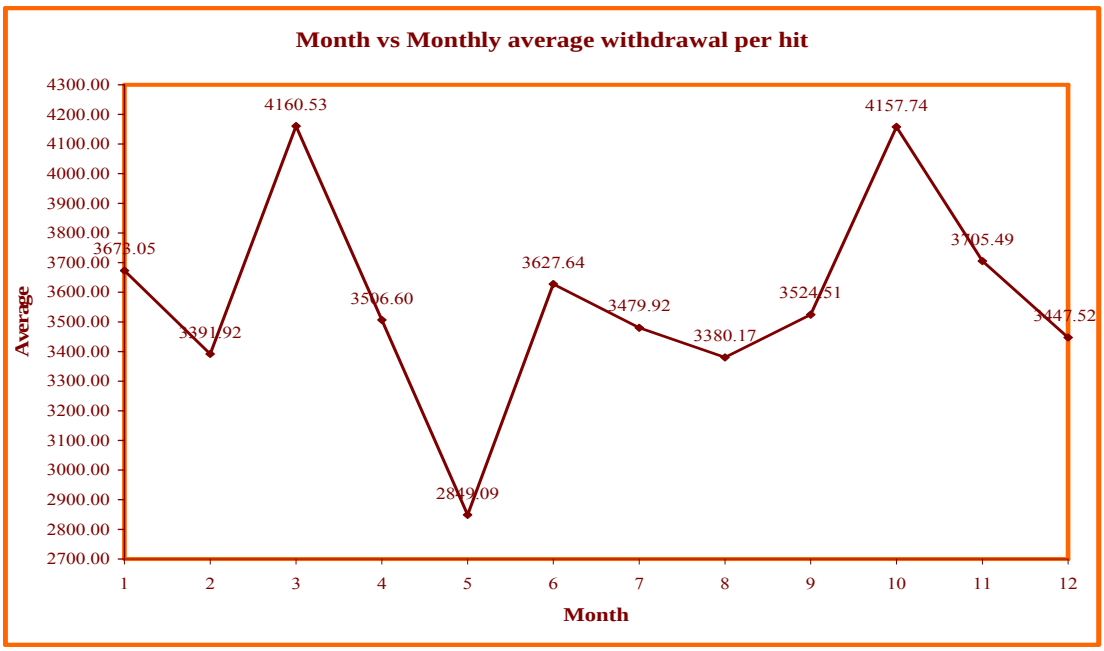
CHART 3.5



CONCLUSION

From chart 3.5 it is clear that month wise total amount withdrawn increases from 1st to 3rd month but from 4th and 5th month, it is decreasing due to specific reason. Again from 5th month to 8th month, it increases in form of straight line. In the month of October, it is increased very fast and again it decreases during 11th and 12th month.

CHART 3.6



3.2.2 METHOD OF MOVING AVERAGES

We have already studied the trend through free hand method. These are very irregular and having ups and down movements of the curves, so we need to study moving average method.

The moving average series would yield a smooth curve. Here we have calculated quarterly moving average using above created data file "mmlytot.txt". This file contains month wise total number of hits (total number of withdrawals per month), total amount of withdrawal and average amount of withdrawal (i.e. total amount of withdrawal / total number of hits).

To read the data file "mmlytot.txt" and create a text file "qmahit.txt" having quarterly moving average of hits and also to generate the well formatted table showing quarterly moving average of hits on the screen, we have prepared C++ programs 'qmahit.cpp' and 'qmaamt.cpp' for quarterly moving average of hits and quarterly moving average of amount withdrawn respectively.

Structure of the output data file qmahit.txt is as under:

Name of the field	Type-Format
Month number	Numeric- integer
Quarterly moving average	Numeric- float

Algorithm for the program 'qmahit.cpp' is given below:

1. Open an input data file1 "mmlytot.txt" to read in "r" mode
2. Open an output data file2 "qmahit.txt" to write in "w" mode
3. Read the file "mmlytot.txt" up to end of file and generate two one dimensional arrays to store month numbers and total number of hits per month. Thus we have initialised two arrays namely mnth[] and hits[] with 12 elements each.
4. To calculate quarterly moving average for the months 2 to 11, repeat following process for i= 2 to 11:
    a. Calculate quarterly moving total, say mtot3[i], Calculate hits[i-1] + hits[i] + hits[i+1] and store the result to mtot3[i].
    b. Calculate quarterly moving average, say mavg3[i], calculate mtot3[i]/3 and store the result to mavg3[i].

c. Write the month number and quarterly moving average to the output file "qmahit.txt" and Print the month number, hits, quarterly moving total and quarterly moving average in formatted way on the screen.

5. Close all data files and terminate the process.

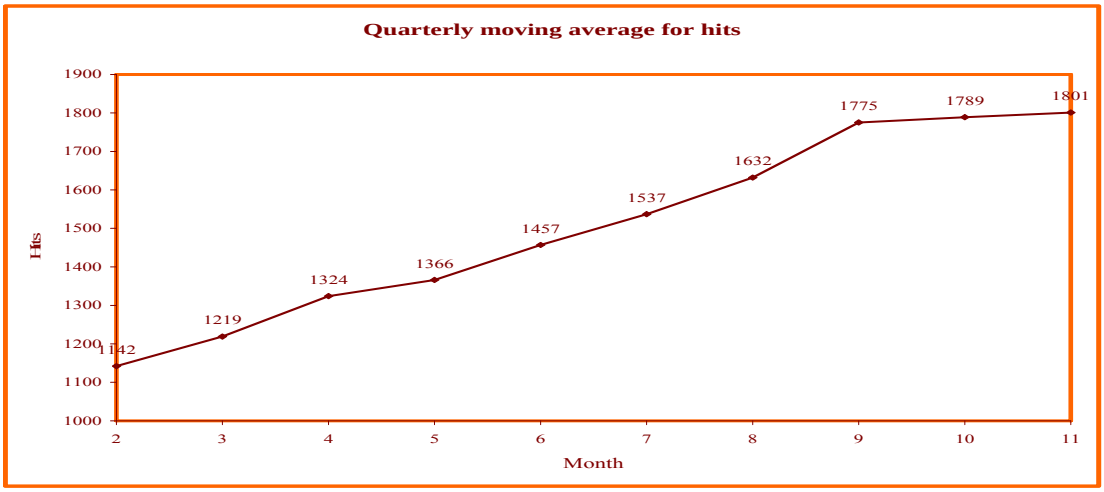
Program 'qmahit.cpp' generates the table of quarterly moving average for month wise number of hits and shows output on screen at figure 3.2:

FIGURE 3.2 Quarterly moving average for hits

Month Number	Hits	Quarterly moving total	Quarterly moving average
1	1039	0	
2	1176	3426	1142.00
3	1211	3659	1219.00
4	1272	3974	1324.00
5	1491	4098	1366.00
6	1335	4372	1457.00
7	1546	4612	1537.00
8	1731	4897	1632.00
9	1620	5327	1775.00
10	1976	5367	1789.00
11	1771	5483	1801.00
12	1656	0	

We have opened the generated text file "qmahit.txt" in Excel, saved the same in Excel format and generated following chart.

CHART 3.7



CONCLUSION

From chart 3.7 it is clear that number of hits increases slowly 2nd month to 5th month, but from 5th month to 9th month it increases rapidly. However 9th to 11th month, number of hits is more or less same but it is in increasing sequence. Further we can say that maximum number of hit is in 3rd and 4th quarter.

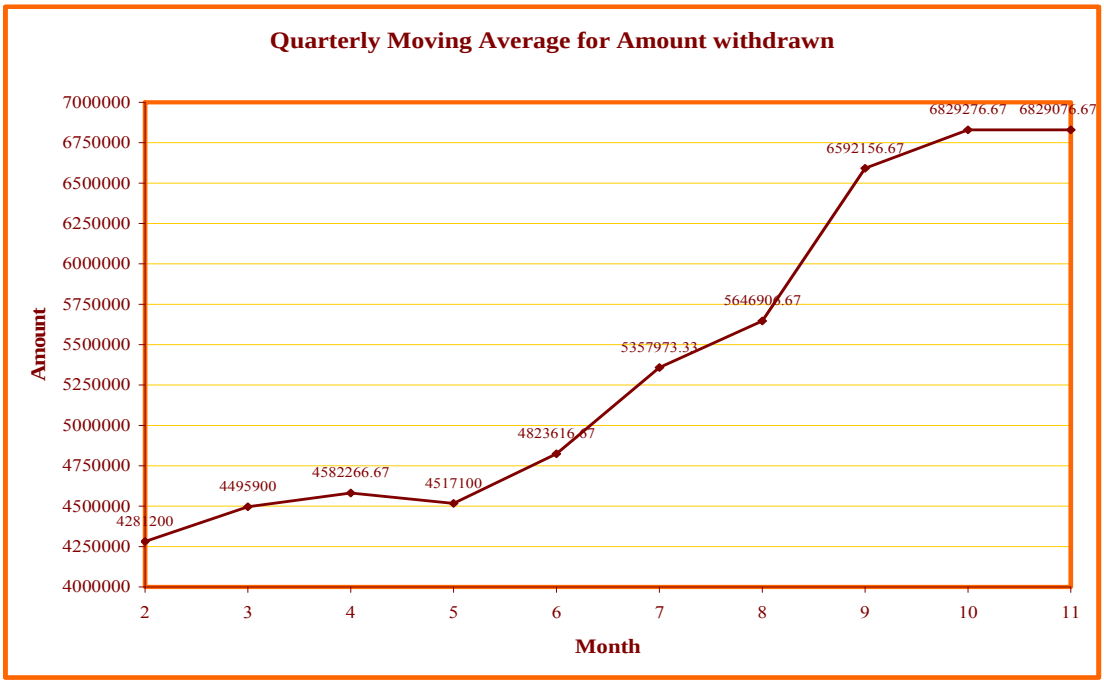
Similarly we have also created C++ program 'qmaamt.cpp' to generate the well formatted table showing quarterly moving average of amount withdrawn.

FIGURE 3.3 Quarterly moving average for amount

Month Number	Transaction of amount During month	Quarterly moving total	Quarterly moving average
1	3816300.00	0.00	
2	3988900.00	12843600.00	4281200.00
3	5038400.00	13487700.00	4495900.00
4	4460400.00	13746800.00	4582266.67
5	4248000.00	13551300.00	4517100.00
6	4842900.00	14470850.00	4823616.67
7	5379950.00	16073920.00	5357973.33
8	5851070.00	16940720.00	5646906.67
9	5709700.00	19776470.00	6592156.67
10	8215700.00	20487030.00	6829276.67
11	6562430.00	20487230.00	6829076.67
12	5709100.00	0.00	

We have also generated one output text file "qmaamt.txt" contains month number and quarterly moving averages. We have opened this file in Excel and saved in Excel format and generated following chart.

CHART 3.8



CONCLUSION

From chart 3.8 it is clear that total amount withdrawn is more or less same from 1st month to 5th month. However from 5th month, total amount withdrawn increase in the form of straight line. This shows that from June to October people need to spend much more money due to several personal and official purposes.

Next we need to discuss method of least square to find the linear trend.

3.2.3 METHOD OF LEAST SQUARES

Under this method, a mathematical relationship is established between the month X and the variable Y i.e quarterly average hits. Here we have considered date 1 to 7 as first quarter, date 8 to 14 as second quarter date 15 to 21 as third quarter and date 22 to end of the month as quarter 4.

A straight line of trend is obtained by using the equation for a straight line, which is $Y_t = a + bX$, Where the value of 'a' is merely the Y-intercept or the height of the line above origin. That is, when X=0, Y=a. The other constant 'b' represents the slope of the trend line. To estimate the value of the constants in the above equation, we start with normal equations. The following are the two normal equations for fitting a straight line by the method of least squares:

$$\sum(y) = na + b\sum(x)$$

$$\sum(xy) = a\sum(x) + b\sum(x^2)$$

3.2.3.1 FITTING OF STRAIGHT LINE

We have seen method of moving averages in above programs 'qmahit.cpp' and 'qmaamt.cpp'. Following is the initial output obtained by the program named 'irrvartn.cpp', showing fitting of straight line. Here X is month number and Y is average number of hits per week.

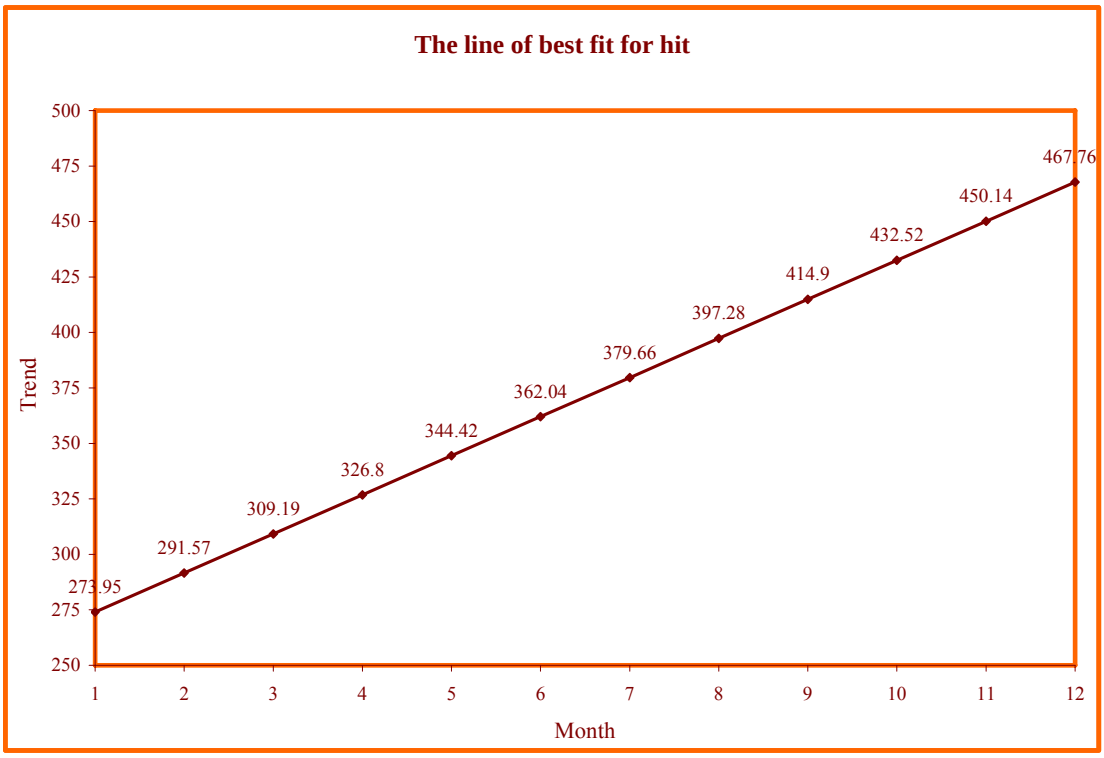
FIGURE 3.4 Weekly total number of hits and trend value

Month		WEEK I	II	III	Last	Total	Average			TrendY
X	Xdev						Y	Xdev^2	XdevY	a+b*xdev
1	-5.50	242	233	213	351	1039	259.75	30.25	-1428.62	274.37
2	-4.50	319	315	285	257	1176	294.00	20.25	-1323.00	292.00
3	-3.50	284	249	240	438	1211	302.75	12.25	-1059.62	309.63
4	-2.50	332	299	261	380	1272	318.00	6.25	-795.00	327.26
5	-1.50	343	358	342	448	1491	372.75	2.25	-559.12	344.89
6	-0.50	370	340	294	331	1335	333.75	0.25	-166.88	362.52
7	0.50	389	363	368	426	1546	386.50	0.25	193.25	380.15
8	1.50	456	368	341	566	1731	432.75	2.25	649.12	397.78
9	2.50	431	359	353	477	1620	405.00	6.25	1012.50	415.41
10	3.50	505	372	346	753	1976	494.00	12.25	1729.00	433.04
11	4.50	419	422	405	525	1771	442.75	20.25	1992.38	450.67
12	5.50	311	364	331	650	1656	414.00	30.25	2277.00	468.29
	0.00					17824	4456.00	143.00	2521.00	

a=371.33 , b=17.63

We have also generated the output data file "leastsq.txt" to store the month wise trend values with two fields like month number and trend value. Opened this file in Excel named 'leastsq.xls' and created chart showing trend line at chart 3.9.

CHART 3.9



CONCLUSION

Here value of b is 17.63 i.e. positive, so the slope will be upwards. It is clearly seen by the figures in the output and in chart, number of hits increases every month.

We have already studied least square method for average number of hits per week. Similarly we have written the program 'irramt.cpp' and following is the initial output obtained by the program named 'irramt.cpp', showing fitting of straight line. Here X is month number and Y is average amount withdrawn per week.

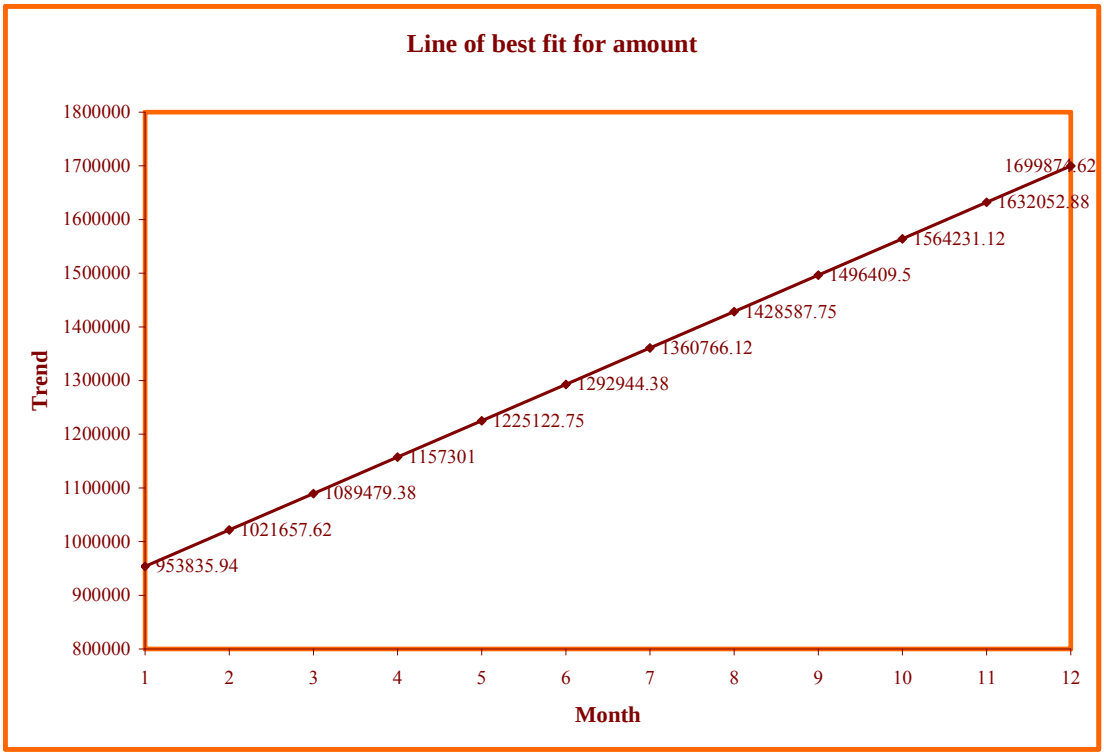
FIGURE 3.5 Least square method for weekly total amount withdrawn

Month		WEEK								
X	Xdev	I	II	III	Last	Weekly Total	Weekly Average Y	Xdev^2	XdevY	TrendY a+b*xdev
1	-5.50	912000	910700	678800	1306800	3808300.00	952075.00	30.25	-5236412.50	953835.83
2	-4.50	1028600	849600	1032600	1063600	3974400.00	993600.00	20.25	-4471200.00	1021657.54
3	-3.50	1225200	1064000	938000	1791100	5018300.00	1254575.00	12.25	-4391012.50	1089479.24
4	-2.50	1188300	898900	817000	1551300	4455500.00	1113875.00	6.25	-2784687.50	1157300.95
5	-1.50	933000	1087400	931100	1286200	4237700.00	1059425.00	2.25	-1589137.50	1225122.65
6	-0.50	1192100	1295300	1111400	1242700	4841500.00	1210375.00	0.25	-605187.50	1292944.36
7	0.50	1561600	1268400	1100450	1446500	5376950.00	1344237.50	0.25	672118.75	1360766.06
8	1.50	1643300	1255600	1085870	1863300	5848070.00	1462017.50	2.25	2193026.25	1428587.77
9	2.50	1488000	1395600	1209800	1610300	5703700.00	1425925.00	6.25	3564812.50	1496409.47
10	3.50	1788400	1449100	1510200	3447500	8195200.00	2048800.00	12.25	7170800.00	1564231.17
11	4.50	1599330	1580200	1632000	1748800	6560330.00	1640082.50	20.25	7380371.25	1632052.88
12	5.50	815100	1219100	1254100	2380800	5669100.00	1417275.00	30.25	7795012.50	1699874.58
	0.00						15922262.50	143.00	9698503.75	

a=1326855.21 b=67821.70

We have also generated the output data file "leastamt.txt" to store the month wise trend values for amount withdrawn with two fields like month number and trend value. Opened this file in Excel named 'leastamt.xls' and created following chart showing trend line.

CHART 3.10



Conclusion:

Here value of b is 67821.70 i.e. positive, so the slope will be upwards. It is clearly seen by the figures in the output and in chart, Amount withdrawn per month increases every month.

3.2.3.2 FITTING OF SECOND DEGREE CURVE

General form for second degree equation is $Y_t = a + bX + cX^2$

$$\sum(y) = na + b\sum(x) + c\sum\left(x^2\right)$$

$$\sum(xy) = a\sum(x) + b\sum(x^2) + c\sum\left(x^3\right)$$

$$\sum(x^2 y) = a\sum(x^2) + b\sum(x^3) + c\sum\left(x^4\right)$$

Following is the initial output obtained by the program named 'lsq2nd.cpp', showing fitting of second degree curve for hits. Here X is month number and Y is average number of hits per week.

FIGURE 3.6 Second degree curve for hits

Month X	Xdev	WEEK I	II	III	Last	Tot	Avg Y	Xdev^2	Xdev^3	Xdev^4	XdevY	Xdev^2y
1	-5.50	242	233	213	351	1039	259.75	30.25	-166.38	915.06	-1428.62	7857.44
2	-4.50	319	315	285	257	1176	294.00	20.25	-91.12	410.06	-1323.00	5953.50
3	-3.50	284	249	240	438	1211	302.75	12.25	-42.88	150.06	-1059.62	3708.69
4	-2.50	332	299	261	380	1272	318.00	6.25	-15.62	39.06	-795.00	1987.50
5	-1.50	343	358	342	448	1491	372.75	2.25	-3.38	5.06	-559.12	838.69
6	-0.50	370	340	294	331	1335	333.75	0.25	-0.12	0.06	-166.88	83.44
7	0.50	389	363	368	426	1546	386.50	0.25	0.12	0.06	193.25	96.62
8	1.50	456	368	341	566	1731	432.75	2.25	3.38	5.06	649.12	973.69
9	2.50	431	359	353	477	1620	405.00	6.25	15.62	39.06	1012.50	2531.25
10	3.50	505	372	346	753	1976	494.00	12.25	42.88	150.06	1729.00	6051.50
11	4.50	419	422	405	525	1771	442.75	20.25	91.12	410.06	1992.38	8965.69
12	5.50	311	364	331	650	1656	414.00	30.25	166.38	915.06	2277.00	12523.50
	0.00					17824	4456.00	143.00	166.38	3038.75	2521.00	51571.50

Using above values simultaneous equations are as under:

$$\begin{aligned} 12a + 0b + 143c &= 4456.00 \\ 0a + 143b + 166.38c &= 2521.00 \\ 143a + 166.38b + 3038.75c &= 51571.50 \end{aligned}$$

From these equations, we have formed a matrix as under:

$$\begin{bmatrix} 12.00 & 0.00 & 143.00 \\ 0.00 & 143.00 & 166.38 \\ 143.00 & 166.38 & 3038.75 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 4456.00 \\ 2521.00 \\ 51571.50 \end{bmatrix}$$

We have developed a program ‘lsq2sol.cpp’, which initialises above matrix of order 3x4, in two dimensional arrays. By the method of reduction, using transformations, we found the values of a, b and c. The output matrix of the program is as under:

```
Command Prompt - tc
   1.000000      0.000000      0.000000    417.934875
   0.000000      1.000000      0.000000     22.179359
   0.000000      0.000000      1.000000     -3.910617
```

This shows that, a= 417.934875, b= 22.179359 and c= -3.910617

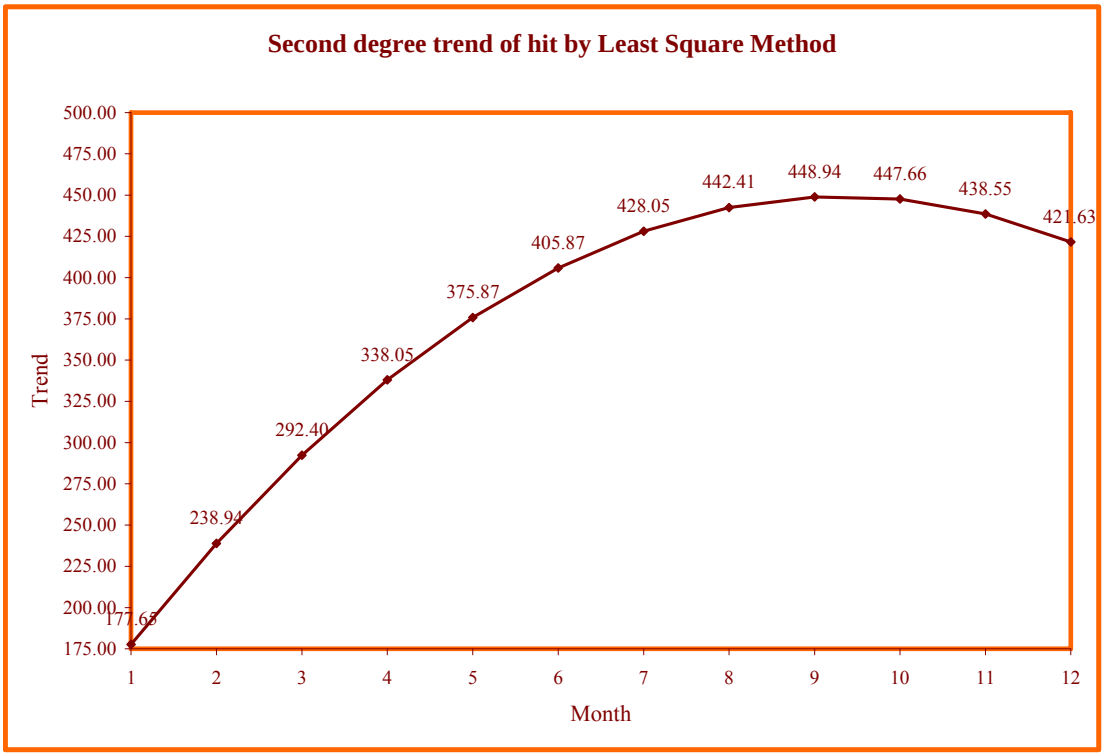
So, the equation of trend line will be $Y = 417.934875 + 22.179359x - 3.910617x^2$

We have calculated the trend values using Excel worksheet ‘lsq2nd.xls’. The resulted table is:

TABLE 3.2

Mnth	X	X^2	Trend
1	-5.5	30.25	177.36
2	-4.5	20.25	238.58
3	-3.5	12.25	291.98
4	-2.5	6.25	337.58
5	-1.5	2.25	375.36
6	-0.5	0.25	405.33
7	0.5	0.25	427.50
8	1.5	2.25	441.85
9	2.5	6.25	448.39
10	3.5	12.25	447.12
11	4.5	20.25	438.05
12	5.5	30.25	421.16

CHART 3.11



CONCLUSION:

From chart 3.11 it is clear that number of hit increases from 1st month to 10th month rapidly. However from 10th month it is started decreasing. Maximum hits are in the month of October.

Similarly we have obtained the initial output by the program named ‘lsq2ndam.cpp’, showing fitting of second degree curve for weekly amount. Here X is month number and Y is average amount withdrawn per quarter (week).

FIGURE 3.7 Fitting of second degree curve for amount withdrawn

Month X	Xdev	WEEK I	II	III	Last	Tot	Avg Y
1	-5.50	912000	910700	678800	1306800	3808300	952075.00
2	-4.50	1028600	849600	1032600	1063600	3974400	993600.00
3	-3.50	1225200	1064000	938000	1791100	5018300	1254575.00
4	-2.50	1188300	898900	817000	1551300	4455500	1113875.00
5	-1.50	933000	1087400	931100	1286200	4237700	1059425.00
6	-0.50	1192100	1295300	1111400	1242700	4841500	1210375.00
7	0.50	1561600	1268400	1100450	1446500	5376950	1344237.50
8	1.50	1643300	1255600	1085870	1863300	5848070	1462017.50
9	2.50	1488000	1395600	1209800	1610300	5703700	1425925.00
10	3.50	1788400	1449100	1510200	3447500	8195200	2048800.00
11	4.50	1599330	1580200	1632000	1748800	6560330	1640082.50
12	5.50	815100	1219100	1254100	2380800	5669100	1417275.00
	0.00					63689050	

Month X	Xdev	Avg Y	Xdev^2	Xdev^3	Xdev^4	XdevY	Xdev^2y
1	-5.50	952075.00	30.25	-166.38	915.06	-5236412.50	28800268.75
2	-4.50	993600.00	20.25	-91.12	410.06	-4471200.00	20120400.00
3	-3.50	1254575.00	12.25	-42.88	150.06	-4391012.50	15368543.75
4	-2.50	1113875.00	6.25	-15.62	39.06	-2784687.50	6961718.75
5	-1.50	1059425.00	2.25	-3.38	5.06	-1589137.50	2383706.25
6	-0.50	1210375.00	0.25	-0.12	0.06	-605187.50	302593.75
7	0.50	1344237.50	0.25	0.12	0.06	672118.75	336059.38
8	1.50	1462017.50	2.25	3.38	5.06	2193026.25	3289539.38
9	2.50	1425925.00	6.25	15.62	39.06	3564812.50	8912031.25
10	3.50	2048800.00	12.25	42.88	150.06	7170800.00	25097800.00
11	4.50	1640082.50	20.25	91.12	410.06	7380371.25	33211670.62
12	5.50	1417275.00	30.25	166.38	915.06	7795012.50	42872568.75
	0.00	15922262.50	143.00	166.38	3038.75	9698503.75	187656900.62

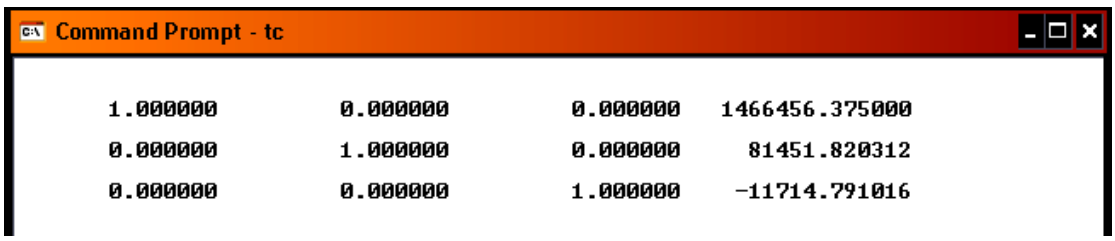
Using above values simultaneous equations are as under:

$$12a + 0b + 143c = 15922262.50$$
$$0a + 143b + 166.38c = 9698503.75$$
$$143a + 166.38b + 3038.75c = 187656900.62$$

From these equations, we have formed a matrix as under:

$$\begin{bmatrix} 12.00 & 0.00 & 143.00 \\ 0.00 & 143.00 & 166.38 \\ 143.00 & 166.38 & 3038.75 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 15922262.50 \\ 9698503.75 \\ 187656900.62 \end{bmatrix}$$

We have developed a program ‘lsq2sola.cpp’, which initialises above matrix of order 3x4, in two dimensional arrays. By the method of reduction, using transformations, we found the values of a, b and c. The output matrix of the program is as under:



This shows that,

a= 1466456.375000, b= 81451.820312, c= -11714.791016

So, the equation of trend line will be

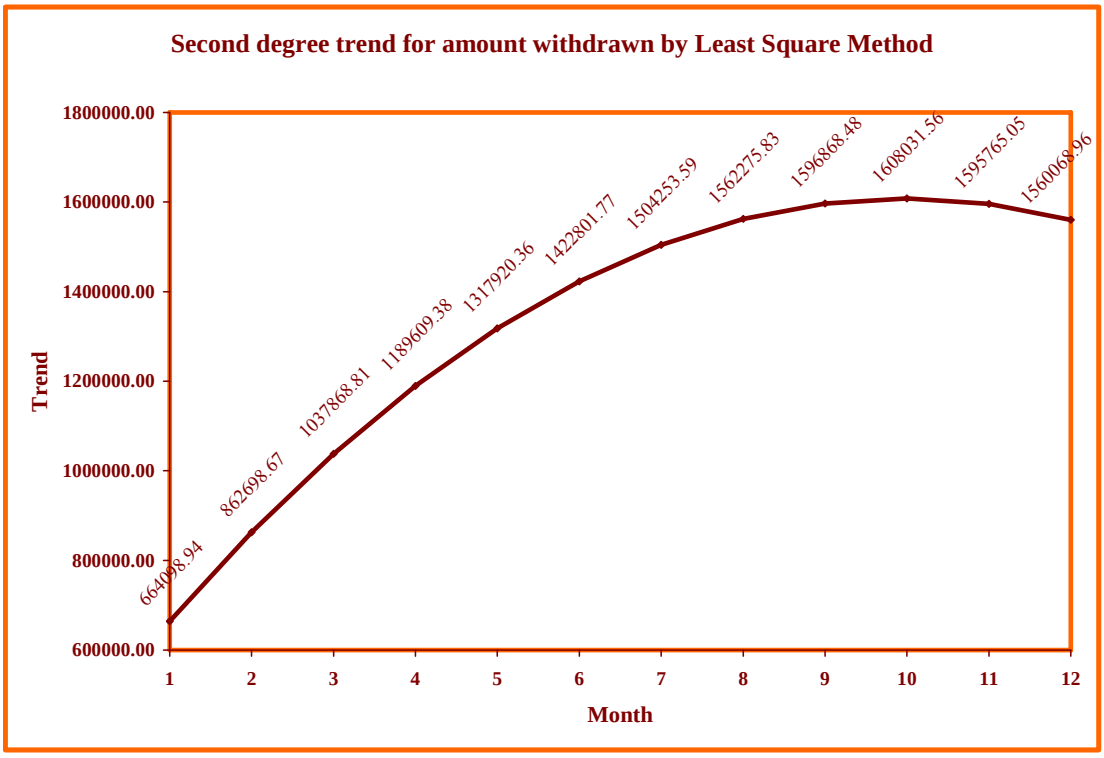
$$Y = 1466456.375000 + 81451.820312x - 11714.791016x^2$$

We have calculated the trend values using Excel. The resulted table is as under:

TABLE 3.3

Month	x	x^2	Trend
1	-5.5	30.25	664098.94
2	-4.5	20.25	862698.67
3	-3.5	12.25	1037868.81
4	-2.5	6.25	1189609.38
5	-1.5	2.25	1317920.36
6	-0.5	0.25	1422801.77
7	0.5	0.25	1504253.59
8	1.5	2.25	1562275.83
9	2.5	6.25	1596868.48
10	3.5	12.25	1608031.56
11	4.5	20.25	1595765.05
12	5.5	30.25	1560068.96

CHART 3.12



CONCLUSION

From chart 3.12 it is clear that amount withdrawn increases from 1st month to 10th month rapidly. However from 10th month it is started decreasing. Maximum amount withdrawn is in the month of October.

3.3 MEASUREMENT OF SEASONAL VARIATIONS

The effects of trend cycles and irregular fluctuations eliminated from the original time series data to obtain an estimate of seasonal variation. We have applied ratio to trend method to deseasonalise the time series. Such deseasonalised data are known as seasonally adjusted data.

3.3.1 RATIO TO TREND METHOD

This method isolates the seasonal factor after eliminating the trend from time series. Trend is eliminated by computing the ratios, and then random elements disappear when the ratios are averaged. We have applied this method for hits as well as amount withdrawn. The steps involved in computation of index for hits are:

5. Determined trend value by the method of least squares, using C++ program 'irrvartn.cpp'. It generates deviation from mean, month wise weekly total number of hits, total number of hits per month and calculations for trend value. The displayed output on screen is as under:

Month		WEEK					Average			TrendY
X	Xdev	I	II	III	Last	Total	Y	Xdev^2	XdevY	a+b*xdev
1	-5.50	242	233	213	351	1039	259.75	30.25	-1428.62	274.37
2	-4.50	319	315	285	257	1176	294.00	20.25	-1323.00	292.00
3	-3.50	284	249	240	438	1211	302.75	12.25	-1059.62	309.63
4	-2.50	332	299	261	380	1272	318.00	6.25	-795.00	327.26
5	-1.50	343	358	342	448	1491	372.75	2.25	-559.12	344.89
6	-0.50	370	340	294	331	1335	333.75	0.25	-166.88	362.52
7	0.50	389	363	368	426	1546	386.50	0.25	193.25	380.15
8	1.50	456	368	341	566	1731	432.75	2.25	649.12	397.78
9	2.50	431	359	353	477	1620	405.00	6.25	1012.50	415.41
10	3.50	505	372	346	753	1976	494.00	12.25	1729.00	433.04
11	4.50	419	422	405	525	1771	442.75	20.25	1992.38	450.67
12	5.50	311	364	331	650	1656	414.00	30.25	2277.00	468.29
	0.00					17824	4456.00	143.00	2521.00	

a=371.33 b=17.63

Thus $Y = 371.33 + 17.63x$,

Where x is measured in months and origin is last week of June or first week of July. From above equation value of Y increases by 17.63 every month, so, Y increases by 17.63/4=4.4075 every week. Following table shows weekly trends by adding 4.4075 every week. Number of hits in the first week of July is 389. For the second week of July add 4.4075 to 389, we get 393.41, for the third week of July add 4.4075 to 393.41, we get 397.81 … and so on.

Similarly,

For the last week of June subtract 4.4075 from 389, we get 384.59, for the third week of June subtract 4.4075 from 384.59, we get 380.19… and so on.

Thus we have generated table showing weekly trend values at figure 3.8.

FIGURE 3.8 Weekly trend values

Month	WEEK I	II	III	Last
1	283.22	287.63	292.04	296.45
2	300.85	305.26	309.67	314.08
3	318.48	322.89	327.30	331.70
4	336.11	340.52	344.93	349.33
5	353.74	358.15	362.56	366.96
6	371.37	375.78	380.19	384.59
7	389.00	393.41	397.81	402.22
8	406.63	411.04	415.44	419.85
9	424.26	428.67	433.07	437.48
10	441.89	446.30	450.70	455.11
11	459.52	463.92	468.33	472.74
12	477.15	481.55	485.96	490.37

6. Divide the original data quarter by quarter (Week as a quarter of the month) by the corresponding trend values and express them as percentages. From above mentioned outputs, in first week of January total number of hits=242, Weekly trend = 283.22

$Percentage = \frac{242}{283.22} \times 100 = 85.44$ and so on for all respective values.

FIGURE 3.9 Weekly percentages of hits

Month	WEEK I	II	III	Last	Total Perc	Average	Adjusted Seasonal Index
1	85.44	81.01	72.94	118.40	357.79	89.45	93.41
2	106.03	103.19	92.03	81.83	383.08	95.77	100.01
3	89.17	77.12	73.33	132.05	371.66	92.92	97.03
4	98.78	87.81	75.67	108.78	371.03	92.76	96.86
5	96.96	99.96	94.33	122.08	413.34	103.33	107.91
6	99.63	90.48	77.33	86.07	353.51	88.38	92.29
7	100.00	92.27	92.51	105.91	390.69	97.67	102.00
8	112.14	89.53	82.08	134.81	418.56	104.64	109.27
9	101.59	83.75	81.51	109.03	375.88	93.97	98.13
10	114.28	83.35	76.77	165.45	439.86	109.96	114.83
11	91.18	90.96	86.48	111.05	379.68	94.92	99.12
12	65.18	75.59	68.11	132.55	341.43	85.36	89.14
						1149.13	

7. Average the different values for a quarter.
8. Adjust all these averages and find the adjusted seasonal index, i.e. multiply each average by 1200/1149.13.

It may be noted here that step (2) eliminates the trend and the values so obtained include cyclical and irregular variations. Step (3) frees the values from cyclical and irregular variations.

FIGURE 3.10 Cyclical irregularities

Month X I	Observed Values Yt II	Trend Values Tt III	Seasonal Index IV	Cyclical Irrelars (%) CtRt V	Three Monthly Moving Totals VI	Three Monthly Moving Averages (%) Ct VII	Irragular Variations (%) CtRt/Ct Rt VIII
1	259.75	274.37	93.41	101.35	0.00	0.00	0.0000
2	294.00	292.00	100.01	100.67	302.80	100.93	99.7433
3	302.75	309.63	97.03	100.77	301.76	100.59	100.1835
4	318.00	327.26	96.86	100.32	301.25	100.42	99.9016
5	372.75	344.89	107.91	100.16	300.23	100.08	100.0805
6	333.75	362.52	92.29	99.76	299.59	99.86	99.8914
7	386.50	380.15	102.00	99.68	299.00	99.67	100.0154
8	432.75	397.78	109.27	99.56	298.59	99.53	100.0289
9	405.00	415.41	98.13	99.35	298.25	99.42	99.9336
10	494.00	433.04	114.83	99.34	297.81	99.27	100.0736
11	442.75	450.67	99.12	99.11	297.64	99.21	99.9013
12	414.00	468.29	89.14	99.18	0.00	0.00	0.0000

We have already studied measurement of seasonal variations for the visit of customer to ATM i.e. number of hits. We have studied the same for the amount withdrawn also. The steps involved in computation of index for amount withdrawn are:

1. Determined trend value by the method of least squares, using C++ program 'irramt.cpp'. The received output is as shown in figure 3.11:

FIGURE 3.11 Weekly total amount withdrawn

Month		WEEK I	II	III	Last	Total
X	Xdev					
1	-5.50	912000	910700	670800	1306800	3800300
2	-4.50	1028600	849600	1032600	1063600	3974400
3	-3.50	1225200	1064000	930000	1791100	5010300
4	-2.50	1188300	898900	817000	1551300	4455500
5	-1.50	933000	1007400	931100	1286200	4237700
6	-0.50	1192100	1295300	1111400	1242700	4841500
7	0.50	1561600	1260400	1100450	1446500	5376950
8	1.50	1643300	1255600	1085870	1063300	5848070
9	2.50	1480000	1395600	1209000	1610300	5703700
10	3.50	1788400	1449100	1510200	3447500	8195200
11	4.50	1597330	1580200	1632000	1748800	6560330
12	5.50	815100	1219100	1254100	2380800	5669100
	0.00					

Month		Weekly Total	Weekly Average			TrendY
X	Xdev		Y	Xdev^2	XdevY	a+b*xdev
1	-5.50	3800300.00	952075.00	30.25	-5236412.50	953035.94
2	-4.50	3974400.00	993600.00	20.25	-4471200.00	1021657.62
3	-3.50	5010300.00	1254575.00	12.25	-4391012.50	1089479.38
4	-2.50	4455500.00	1113875.00	6.25	-2784687.50	1157301.00
5	-1.50	4237700.00	1059425.00	2.25	-1589137.50	1225122.75
6	-0.50	4841500.00	1210375.00	0.25	-605187.50	1292944.38
7	0.50	5376950.00	1344237.50	0.25	672118.75	1360766.12
8	1.50	5848070.00	1462017.50	2.25	2193026.25	1428587.75
9	2.50	5703700.00	1425925.00	6.25	3564812.50	1496409.50
10	3.50	8195200.00	2048800.00	12.25	7170800.00	1564231.12
11	4.50	6560330.00	1640082.50	20.25	7380371.00	1632052.88
12	5.50	5669100.00	1417275.00	30.25	7795012.50	1699874.62
	0.00		15922263.00	143.00	9698502.00	

a=1326855.25 , b=67821.70

Thus $Y = 1326855.25 + 67821.70x$,

Where x is measured in months and origin is last week of June or first week of July. From above equation value of Y increases by 67821.70 every month, so, Y increases by 67821.70/4=16955.425 every week. Following table shows weekly trends by adding 16955.425 every week.

Amount withdrawn in the first week of July is 1561600. For the second week of July add 16955.43 to 1561600, we get 1578555.43, for the third week of July add 16955.43 to 1578555.43, we get 1595510.85 … and so on.

Similarly, for the last week of June subtract 16955.43 from 1561600, we get 1544644.57, For the third week of June subtract 16955.43 from 1544644.57, we get 1527689.15… and so on.

Thus we have generated following table showing weekly trend values.

FIGURE 3.12 Weekly trend of amount withdrawn

Month	WEEK I	II	III	Last
1	1154669.77	1171625.20	1188580.62	1205536.05
2	1222491.48	1239446.90	1256402.33	1273357.76
3	1290313.18	1307268.61	1324224.03	1341179.46
4	1358134.89	1375090.31	1392045.74	1409001.16
5	1425956.59	1442912.02	1459867.44	1476822.87
6	1493778.30	1510733.72	1527689.15	1544644.57
7	1561600.00	1578555.43	1595510.85	1612466.28
8	1629421.70	1646377.13	1663332.56	1680287.98
9	1697243.41	1714198.84	1731154.26	1748109.69
10	1765065.11	1782020.54	1798975.97	1815931.39
11	1832886.82	1849842.24	1866797.67	1883753.10
12	1900708.52	1917663.95	1934619.38	1951574.80

2. Divide the original data quarter by quarter by the corresponding trend values and express them as percentages.

For first week of January,

Total amount withdrawn =912000, Weekly trend =1154669.77

$$Percentage = \frac{912000}{1154669.77} \times 100$$

$$= 78.98$$

FIGURE 3.13 Weekly percentage of amount withdrawn

Month	WEEK I	II	III	Last	Total Perc	Average	Adjusted Seasonal Index
1	78.98	77.73	57.11	108.48	322.22	80.56	94.78
2	84.14	68.55	82.19	83.53	318.40	79.60	93.66
3	94.95	81.39	70.83	133.55	380.73	95.18	111.99
4	87.49	65.37	58.69	110.10	321.66	80.41	94.62
5	65.43	75.36	63.78	87.09	291.66	72.92	85.79
6	79.80	85.74	72.75	80.45	318.75	79.69	93.76
7	100.00	80.35	68.97	89.71	339.03	84.76	99.73
8	100.85	76.26	65.28	110.89	353.29	88.32	103.92
9	87.67	81.41	69.88	92.12	331.09	82.77	97.39
10	101.32	81.32	83.95	189.85	456.44	114.11	134.26
11	87.26	85.42	87.42	92.84	352.94	88.23	103.82
12	42.88	63.57	64.82	121.99	293.27	73.32	86.27
						1019.87	

3. Find the average of weekly percentages.

4. Adjust all these averages and find the adjusted seasonal index, i.e. multiply each average by 1200/1019.87, Above table shows that 1019.87 is the total of averages..

It may be noted here that step (2) eliminates the trend and the values so obtained include cyclical and irregular variations. Step (3) frees the values from cyclical and irregular variations.

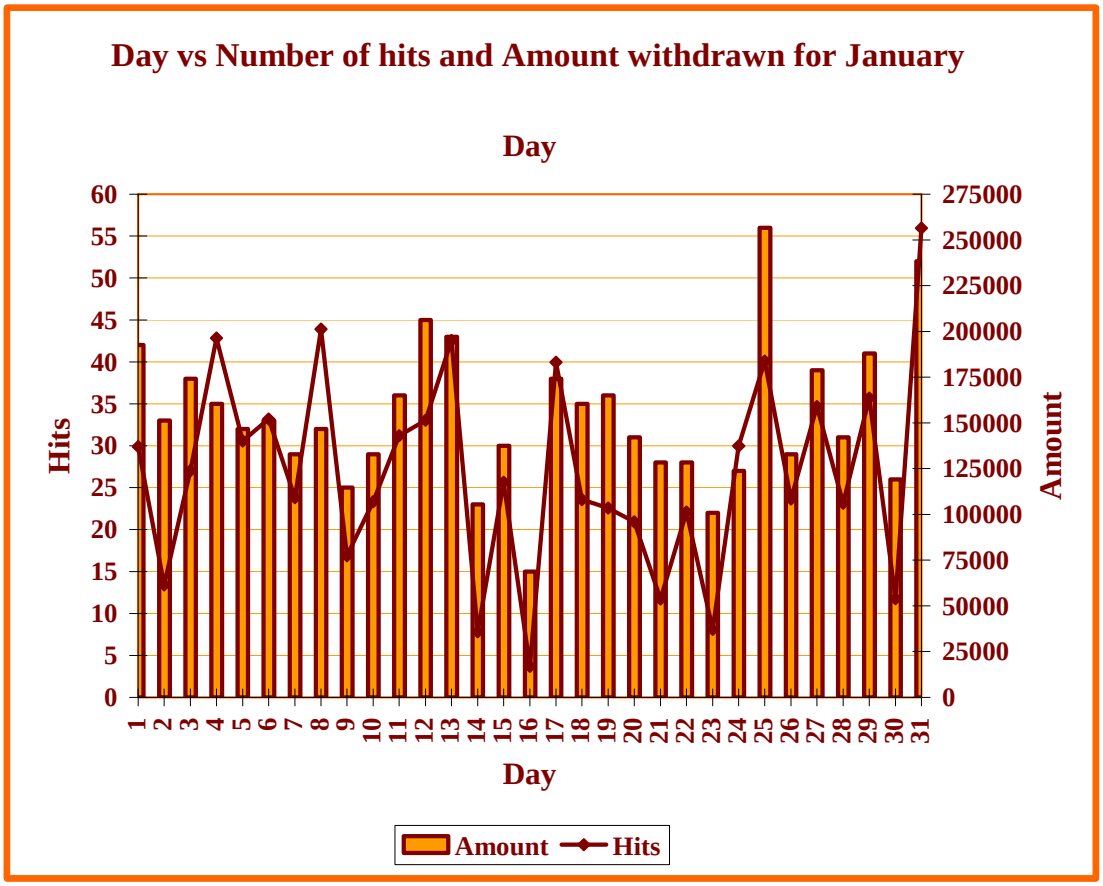
FIGURE 3.14 Cyclical Irregularities in amount withdrawn

Month X I	Observed Values Yt II	Trend Values Tt III	Seasonal Index IV	Cyclical Irrelare (%) CtRt V	Three Monthly Moving Totals VI	Three Monthly Moving Averages (%) Ct VII	Irregular Variatione (%) CtRt/Ct Rt VIII
1	952075.00	953835.83	94.78	105.31	0.00	0.00	0.0000
2	993600.00	1021657.54	93.66	103.84	311.97	103.99	99.8539
3	1254575.00	1089479.24	111.99	102.82	308.38	102.79	100.0272
4	1113875.00	1157300.95	94.62	101.72	305.34	101.78	99.9449
5	1059425.00	1225122.65	85.79	100.79	302.36	100.79	100.0066
6	1210375.00	1292744.36	93.76	99.84	299.69	99.90	99.9458
7	1344237.50	1360766.06	99.73	99.05	297.38	99.13	99.9293
8	1462017.50	1428587.77	103.92	98.48	295.37	98.46	100.0194
9	1425925.00	1496409.47	97.39	97.84	293.87	97.96	99.8824
10	2048000.00	1564231.17	134.26	97.55	292.19	97.40	100.1605
11	1640062.50	1632052.88	103.82	96.80	291.80	97.00	99.7986
12	1417275.00	1699874.58	86.27	96.65	0.00	0.00	0.0000

3.4 COMPARISON OF HITS AND AMOUNT WITHDRAWN

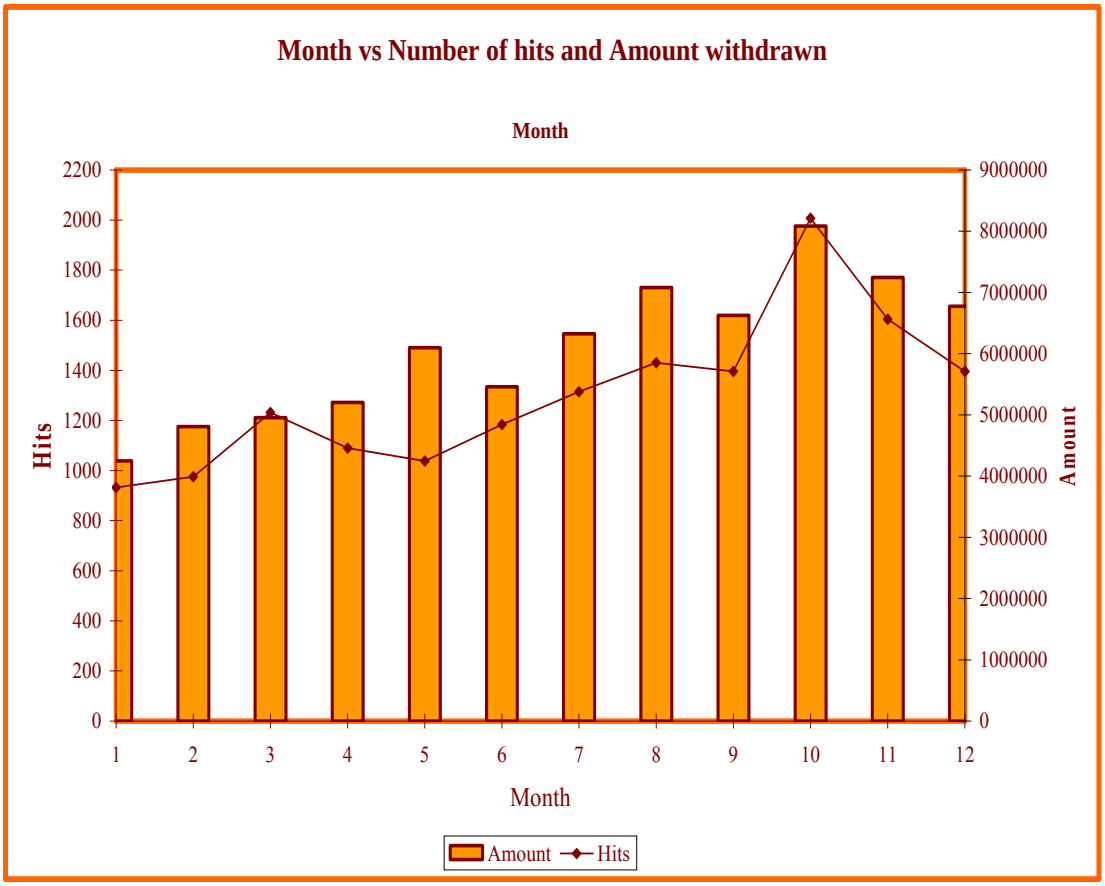
We have already studied the analysis of hits and amount withdrawn individually graphically also. Now let analyse both simultaneously graphically.

CHART 3.13 Day Vs. Number of hits and amount withdrawn for january



From chart 3.13 it is seen that first two week and last two week the amount of withdrawl is more compared to third week. This may be due to reason that in the beginning and at the end money is required for marketing, tuition fees, electricity, telephone bills etc. There is no perfect relation between number of hits and amount withdrawn.

CHART 3.14 Month Vs. number of hits and amount withdrawn



From chart 3.14 it is seen that amount withdrawn is more in the month of August, October and November. This may be due to festival like Janmashtami, Navaratri and new year where more money require for marketing. May is vacation time and hence they go for tour and hence more money is also withdrawn in this month. In monthly withdrawn it seems there is relation between number of hits and amount withdrawn as number of hits increases amount withdrawn is also increases (except march and April).

3.5 ANOVA FOR HITS

We have developed a C++ program named 'anovahit.cpp' to generate ANOVA table for hits. The output of this program is shown here at figure 3.15.

FIGURE 3.15 Month wise weekly hits and sum of squares

	WEEK					
Month	I	II	III	Last	Row Sum	Sum of square
1	242	233	213	351	1039.00	1079521.00
2	319	315	285	257	1176.00	1382976.00
3	284	249	240	438	1211.00	1466521.00
4	332	299	261	380	1272.00	1617984.00
5	343	358	342	448	1491.00	2223081.00
6	370	340	294	331	1335.00	1782225.00
7	389	363	368	426	1546.00	2390116.00
8	456	368	341	566	1731.00	2996361.00
9	431	359	353	477	1620.00	2624400.00
10	505	372	346	753	1976.00	3904576.00
11	419	422	405	525	1771.00	3136441.00
12	311	364	331	650	1656.00	2742336.00
Column Sum	4401	4042	3779	5602	17824.00	27346536.00
Sum of Squ.	19368800	16337764	14280841	31382404	81369808.00	

CF = 6618645.50 Month SS = 217988.50 Week SS = 162171.83
Sum of Hit^2 = 7127382.00 Total SS = 508736.50 Error SS= 128576.17

Analysis of Variance (ANOVA) table

Source of variation	Sum of squares	Degrees of freedom	Mean squares	F-Ratios	
Between weeks(ssc)	162171.83	3	54057.28	13.87	**
Between months(ssr)	217988.50	11	19817.14	5.09	*
Error(sse)	128576.17	33	3896.25		

H_0: Each week has same number of hits.

H_1: Each week has different number of hits.

Conclusion: Here test is significant and hence H_0 is rejected, that is, number of hits per week are different.

3.6 ANOVA FOR AMOUNT WITHDRAWN

We have developed a C++ program named 'anovaamt.cpp' to generate ANOVA table for amount withdrawn. The output of this program is shown here at figure 3.16.

FIGURE 3.16 Month wise amount withdrawn and sum of squares

	WEEK					
Month	I	II	III	Last	Row Sum	Row SS
1	912000	910700	678800	1306800	3808300	1.45e+13
2	1028600	849600	1032600	1063600	3974400	1.58e+13
3	1225200	1064000	938000	1791100	5018300	2.52e+13
4	1188300	898900	817000	1551300	4455500	1.99e+13
5	933000	1087400	931100	1286200	4237700	1.80e+13
6	1192100	1295300	1111400	1242700	4841500	2.34e+13
7	1561600	1268400	1100450	1446500	5376950	2.89e+13
8	1643300	1255600	1085870	1863300	5848070	3.42e+13
9	1488000	1395600	1209800	1610300	5703700	3.25e+13
10	1788400	1449100	1510200	3447500	8195200	6.72e+13
11	1599330	1580200	1632000	1748800	6560330	4.30e+13
12	815100	1219100	1254100	2380800	5669100	3.21e+13
C Sum	15374930	14273900	13301320	20738900	63689050	3.55e+14
C SS	2.36e+14	2.04e+14	1.77e+14	4.30e+14	1.05e+15	

CF=84506147706302 Month SS=4172271845273 Week SS=2757167349306
Sum of amt^2=94404441928300 Total SS=9898294221998 Error SS=2968855027419

Analysis of Variance (ANOVA) table

Source of variation	Sum of squares	Degrees of freedom	Mean squares	F-Ratios	
Between weeks(ssc)	2757167349306	3	919055783102.09	10.22	**
Between months(ssr)	4172271845273	11	379297440479.36	4.22	*
Error(sse)	2968855027419	33	89965303861.17		

H_0: Amount withdrawn per week is same.

H_1: Amount withdrawn per week is different.

Conclusion: Here test is significant and hence H_0 is rejected, that is, amount withdrawn per week are different.

3.7 MEAN AND MODE FOR HITS AND AMOUNT WITHDRAWN

As we know that all the months of year are not having equal numbers of days, so we have grouped this data with equal class interval of 30 days to find the mean.

Sr	Days			No. of hits	Amount withdrawn
1	1	-	30	987	3559800.00
2	31	-	60	1273	4421800.00
3	61	-	90	1166	4862000.00
4	91	-	120	1272	4460400.00
5	121	-	150	1452	4125700.00
6	151	-	180	1331	4832700.00
7	181	-	210	1512	5279250.00
8	211	-	240	1629	5507770.00
9	241	-	270	1614	5616200.00
10	271	-	300	1786	7141400.00
11	**301**	**-**	**330**	**1910**	**7435530.00**
12	331	-	360	1499	5043700.00
			Total	17431	62286250.00
			Daily average	48	173017.36
			Monthly average	1453	5190520.83

Mean:

		For hits	**For amount withdrawn**
n	$=\sum f$	=17431	=62286250.00
Daily Mean	$=\frac{\sum f}{n}$	$=\frac{17431}{360}$=48	$=\frac{62286250.00}{360}$=173017.36
Monthly Mean	$=\frac{\sum f}{n}$	$=\frac{17431}{12}$=1453	$=\frac{62286250.00}{12}$=5190520.83

Mode:

Using grouped totals (monthly): Here maximum number of hits 1910 is in the class 301-330, which therefore is the mode of hits. Maximum amount withdrawn Rs. 7435530 is in the class 301-330, which therefore is the mode of amount withdrawn. (301-330 i.e. around October-November).

Using daily totals: Here maximum number of hits per day is 108 on 31-10-2005, which is therefore is the mode of hits and maximum amount withdrawn Rs.523100 is also on 31-10-2005, which therefore is the mode of amount withdrawn per day.

	Hits	Amount withdrawn
Total	17824	63822850
Maximum	108	523100

This shows that bank must keep amount of Rs. 523100 per day and Rs. 7435530 per month. Again from chart 3.11 and 3.12 it is clear that number of hits and amount withdrawn increase from 1st month to 10th month rapidly. However from 10th month it is started decreasing. Maximum hits and amount withdrawn are in the month of October.

If it is considered then it assured that customer will not return without collecting money from the ATM. If money is kept less than those values then some customers will return without taking money.

Chapter 4

Effect of Monthly and Daily Rest Basis on EMI for Housing, Personal and Car loans

4.1 INTRODUCTION

When one takes a loan, a natural question that comes to mind is how much the EMI (Equated Monthly Instalment) would be that one has to pay back to the bank every month. Finding out the EMI for different tenures allows one to select the best tenure based on one's current and projected income and expenses and possibly other factors.

We have already generated the formula to calculate EMI in the chapter of introduction as: **$EMI = P \times r \times (1 + r)^n / ((1 + r)^n - 1)$**

In this chapter we have calculated EMIs for different rate of interests and different tenures on monthly and daily rest basis and studied the effect. First we discuss the EMI for housing loan.

4.2 HOUSING LOAN

4.2.1 EMI ON REDUCING MONTHLY BASIS

To calculate EMI on monthly rest basis, we prepared a C++ program 'emimmly.cpp' to generate EMIs for a range of terms like 5 years, 10 years, 15 years as well as 20 years where interest rate ranging from 7% to 15% after every intervals of 0.25, i.e. 7.00, 7.25, 7.50, 7.75, 8.00, 8.25, … 15.00 for the amount of Rs.1000000.00.

Let us understand the calculation of EMI by taking one example:

Now, $EMI = P \times r \times (1 + r)^n / ((1 + r)^n - 1)$, Let P=10,00,000, R=7% and N=5 years.

r=7/12 %=7/1200=0.005833, since interest is compounded monthly, i.e. 12 times in a year, so n=N*12=5*12=60

So, EMI $= 1000000 \times 0.005833 \times \dfrac{(1+0.005833)^{60}}{((1+0.005833)^{60}-1)} = 19801.20$

Similarly we have calculated all the EMIs. The output of this program is shown at figure 4.1.

FIGURE 4.1 Home loan EMIs for Rs.10,00,000.00 (Reducing monthly basis)

Rate	5 Yrs	10 Yrs	15 Yrs	20 Yrs
7.00	19801.20	11610.85	8988.28	7752.99
7.25	19919.36	11740.10	9128.63	7903.76
7.50	20037.95	11870.18	9270.12	8055.93
7.75	20156.96	12001.06	9412.76	8209.49
8.00	20276.39	12132.76	9556.52	8364.40
8.25	20396.25	12265.26	9701.40	8520.66
8.50	20516.53	12398.57	9847.40	8678.23
8.75	20637.23	12532.67	9994.49	8837.11
9.00	20758.36	12667.58	10142.67	8997.26
9.25	20879.90	12803.27	10291.92	9158.67
9.50	21001.86	12939.76	10442.25	9321.31
9.75	21124.24	13077.02	10593.63	9485.17
10.00	21247.04	13215.07	10746.05	9650.22
10.25	21370.26	13353.90	10899.51	9816.43
10.50	21493.90	13493.50	11053.99	9983.80
10.75	21617.95	13633.87	11209.48	10152.29
11.00	21742.42	13775.00	11365.97	10321.88
11.25	21867.31	13916.89	11523.45	10492.56
11.50	21992.61	14059.54	11681.90	10664.30
11.75	22118.32	14202.95	11841.31	10837.07
12.00	22244.45	14347.09	12001.68	11010.86
12.25	22370.99	14491.99	12162.99	11185.65
12.50	22497.94	14637.62	12325.22	11361.41
12.75	22625.30	14783.98	12488.37	11538.12
13.00	22753.07	14931.07	12652.42	11715.76
13.25	22881.26	15078.89	12817.36	11894.31
13.50	23009.85	15227.43	12983.19	12073.75
13.75	23138.85	15376.68	13149.87	12254.05
14.00	23268.25	15526.64	13317.41	12435.21
14.25	23398.06	15677.31	13485.80	12617.19
14.50	23528.28	15828.68	13655.01	12799.98
14.75	23658.90	15980.74	13825.04	12983.55
15.00	23789.93	16133.50	13995.87	13167.90

Here, it shows that EMI varies according to interest rate and term. A person can decide the term according to ones repayment capacity as shown above.

4.2.2 EMI ON REDUCING DAILY BASIS

Prepared another C++ program ‘emidaily.cpp’ to generate EMIs on daily rest basis for a range of terms like 5 years, 10 years, 15 years as well as 20 years and interest rate ranging from 7 to 10 after every intervals of 0.25, i.e. 7.00, 7.25, 7.50, 7.75, 8.00, 8.25, … 15.00 for the amount of Rs.1000000.00.

Let us understand the calculation of EMI by taking one example:

Now, EMI = $P \times r \times (1 + r)^n / ((1 + r)^n - 1)$, Let P=10,00,000, R=7% and N=5 years.

r=7/365 %=7/36500=0.000191781, since interest is compounded daily i.e. 365 times in a year, so n=N*365=5*365=1825

So, EMI for a day $= 1000000 \times 0.000191781 \times \frac{(1+0.000191781)^{1825}}{\left((1+0.000191781)^{1825}-1\right)}$

$= 649.4698$

So, EMI for a month = 649.4698 * (365/12) = 19754.71

Similarly we have calculated all the EMIs. The output of this program is at figure 4.2.

FIGURE 4.2 Home loan EMIs for Rs.10,00,000.00 (reducing daily basis)

Rate	5 Yrs	10 Yrs	15 Yrs	20 Yrs
7.00	19754.71	11588.30	8973.99	7742.99
7.25	19871.25	11716.82	9113.92	7893.52
7.50	19988.22	11846.16	9255.01	8045.46
7.75	20105.61	11976.33	9397.25	8198.80
8.00	20223.43	12107.31	9540.63	8353.51
8.25	20341.68	12239.10	9685.13	8509.57
8.50	20460.36	12371.71	9830.76	8666.96
8.75	20579.46	12505.13	9977.50	8825.66
9.00	20698.99	12639.34	10125.33	8985.65
9.25	20818.95	12774.36	10274.25	9146.90
9.50	20939.32	12910.18	10424.25	9309.41
9.75	21060.13	13046.79	10575.32	9473.13
10.00	21181.35	13184.19	10727.44	9638.06
10.25	21303.00	13322.38	10880.61	9804.17
10.50	21425.07	13461.34	11034.81	9971.44
10.75	21547.57	13601.09	11190.03	10139.84
11.00	21670.48	13741.61	11346.26	10309.35
11.25	21793.82	13882.90	11503.48	10479.96
11.50	21917.58	14024.95	11661.70	10651.64
11.75	22041.76	14167.77	11820.89	10824.37
12.00	22166.35	14311.34	11981.04	10998.12
12.25	22291.37	14455.67	12142.14	11172.88
12.50	22416.80	14600.75	12304.18	11348.62
12.75	22542.65	14746.56	12467.15	11525.32
13.00	22668.91	14893.12	12631.02	11702.96
13.25	22795.60	15040.41	12795.81	11881.51
13.50	22922.69	15188.44	12961.48	12060.97
13.75	23050.20	15337.19	13128.02	12241.30
14.00	23178.13	15486.65	13295.43	12422.48
14.25	23306.47	15636.84	13463.70	12604.50
14.50	23435.22	15787.73	13632.80	12787.34
14.75	23564.38	15939.33	13802.73	12970.97
15.00	23693.96	16091.64	13973.47	13155.37

Here, both figure 4.1 and figure 4.2 clearly indicates that amount of EMI with reducing daily rest basis is less than the EMI with reducing monthly rest basis.

4.2.3 DIFFERENCE OF EMI ON REDUCING MONTHLY AND DAILY BASIS

To see the clear difference of EMI amount for daily and monthly reducing basis, prepared a C++ program ‘emidiff.cpp’. The output is at figure 4.3.

FIGURE 4.3

Home loan of Rs.1000000/-

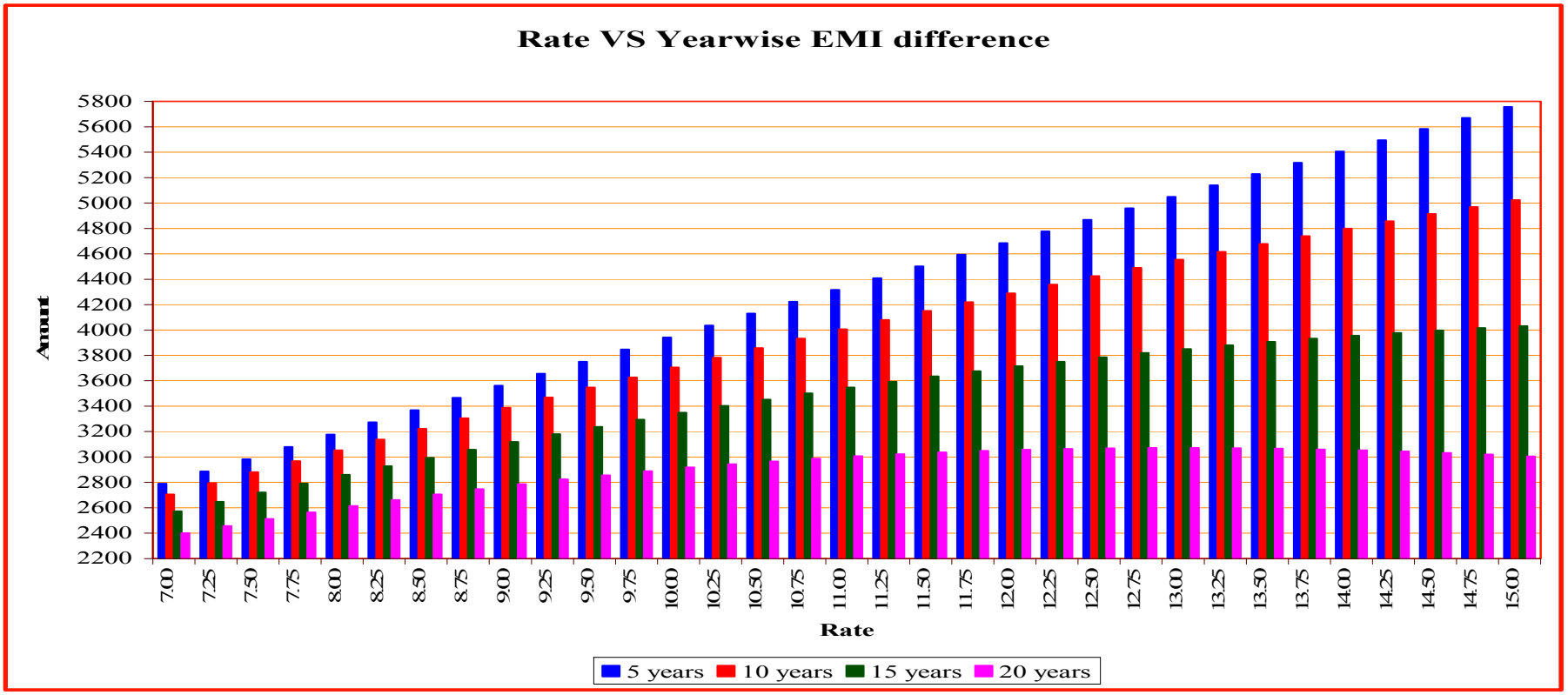
Statement showing EMIs on reducig monthly basis, daily basis and difference

	5 Yrs			10 Yrs			15 Yrs			20 Yrs		
Rate	mnthly	daily	diff	mnthly	daily	diff	mnthly	daily	diff	mnthly	daily	diff
7.00	19801	19755	2790	11611	11588	2706	8988	8974	2573	7753	7743	2401
7.25	19919	19871	2887	11740	11717	2794	9129	9114	2648	7904	7894	2458
7.50	20038	19988	2984	11870	11846	2882	9270	9255	2721	8056	8045	2513
7.75	20157	20106	3081	12001	11976	2968	9413	9397	2792	8209	8199	2565
8.00	20276	20223	3178	12133	12107	3054	9557	9541	2861	8364	8354	2615
8.25	20396	20342	3274	12265	12239	3139	9701	9685	2929	8521	8510	2662
8.50	20517	20460	3370	12399	12372	3223	9847	9831	2994	8678	8667	2706
8.75	20637	20579	3466	12533	12505	3306	9994	9977	3058	8837	8826	2748
9.00	20758	20699	3562	12668	12639	3388	10143	10125	3120	8997	8986	2787
9.25	20880	20819	3657	12803	12774	3469	10292	10274	3181	9159	9147	2824
9.50	21002	20939	3752	12940	12910	3549	10442	10424	3239	9321	9309	2857
9.75	21124	21060	3847	13077	13047	3628	10594	10575	3295	9485	9473	2889
10.00	21247	21181	3941	13215	13184	3706	10746	10727	3350	9650	9638	2918
10.25	21370	21303	4036	13354	13322	3783	10900	10881	3403	9816	9804	2944
10.50	21494	21425	4130	13494	13461	3859	11054	11035	3453	9984	9971	2967
10.75	21618	21548	4223	13634	13601	3933	11209	11190	3502	10152	10140	2988
11.00	21742	21670	4316	13775	13742	4007	11366	11346	3548	10322	10309	3007
11.25	21867	21794	4409	13917	13883	4080	11523	11503	3593	10493	10480	3023
11.50	21993	21918	4502	14060	14025	4151	11682	11662	3636	10664	10652	3037
11.75	22118	22042	4594	14203	14168	4221	11841	11821	3677	10837	10824	3049
12.00	22244	22166	4686	14347	14311	4290	12002	11981	3716	11011	10998	3058
12.25	22371	22291	4777	14492	14456	4358	12163	12142	3752	11186	11173	3065
12.50	22498	22417	4868	14638	14601	4425	12325	12304	3787	11361	11349	3069
12.75	22625	22543	4959	14784	14747	4490	12488	12467	3821	11538	11525	3072
13.00	22753	22669	5049	14931	14893	4554	12652	12631	3852	11716	11703	3072
13.25	22881	22796	5140	15079	15040	4617	12817	12796	3881	11894	11882	3071
13.50	23010	22923	5229	15227	15188	4679	12983	12961	3908	12074	12061	3067
13.75	23139	23050	5318	15377	15337	4740	13150	13128	3933	12254	12241	3061
14.00	23268	23178	5407	15527	15487	4799	13317	13295	3956	12435	12422	3054
14.25	23398	23306	5496	15677	15637	4857	13486	13464	3978	12617	12605	3045
14.50	23528	23435	5584	15829	15788	4914	13655	13633	3998	12800	12787	3033
14.75	23659	23564	5671	15981	15939	4969	13825	13803	4016	12984	12971	3021
15.00	23790	23694	5758	16133	16092	5023	13996	13973	4032	13168	13155	3006

From figure 4.3 we conclude that daily reducing basis is less EMI than monthly reducing basis. For the tenure of 5 years the difference of EMI is ranging from Rs.2790 to Rs.5758, for the tenure of 10 years the difference of EMI is ranging from 2706 to 5023, for the tenure of 15 years the difference of EMI is ranging from 2573 to 4032 and for the tenure of 20 years the difference of EMI is ranging from 2401 to 3006 on monthly repayment of the loan.

We draw the chart for Rate of interest Vs. yearwise EMI difference and shown at chart number 4.1.

CHART 4.1



CONCLUSION

From above chart 4.1 it is clear that this difference of EMI (total amount repaid) increase as rate of interest increases for certain tenure and difference of EMI (total amount repaid) decrease as tenure increases for certain rate of interest.

4.2.4 RATE OF INTEREST AND EMI OF VARIOUS BANKS FOR DIFFERENT PERIODS

For the academic interest, we have collected home loan interest rates and EMI amount (As on 01-03-2010) of various banks from the web site ecompare.co.in. We have also visited banks of surrounding area to verify the rates and EMIs. As our interest is only to know the amount of EMI at certain rate of interest and certain tenure, the lender bank and the effective date is not important for us. Following tables show interest rates and EMI of various banks for different tenure like 5 years, 10 years, 15 years and 20 years. That will be useful to compare our program generated EMIs:

TABLE 4.1 Compilation of Resident Home Loan Search Results for Rs. 10,00,000
Updated on 01 March 2010

Financial Institution	5 years		10 years		15 years		20 years	
	Rate	EMI	Rate	EMI	Rate	EMI	Rate	EMI
Dena Bank	9.75	21124	10.25	13354	10.25	10900	10.25	9816
Punjab & Sind Bank	9.00	20758	11	13775	11	11366	11.00	10322
Federal Bank	10.00	21494	11	13775	11	11366	11.00	10322
Allahabad bank	9.25	20880	9.25	12803	9.25	10292	9.25	9159
Indian Bank	10.50	21494	11	13775				
United Bank	8.75	20637	8.75	12533				
Bank of Baroda	9.50	21002	9.75	13077	10.00	10746		
Syndicate Bank	10.75	21618	11.75	14203				
Punjab National Bank	9.25	20880	10.00	13215	10.50	11054	10.50	9984
Indian Overses Bank	9.00	20758	9.50	12940				
UCO Bank	11.75	22118	11.75	14203	12.00	12002	12.00	11011
Bank of India								
Corporation Bank	11.00	21742	11	13775	11	11366	11	10322
Union Bank of India	9.50	21002						
IDBI Bank	11.25	21867						
Central Bank of India	10.00	21247						
Vijaya Bank	10.75	21618						
State Bank of India	10.75	21618	10.75	13634				
Canara Bank								
J & K Bank	10.25	21370	11	13775				
Andhra Bank								
Karur Vysya Bank	12.00	22244	12.50	14638	13.00	12652		
South indian Bank	12.50	22498	13.5	15227	12.75	12488	12.75	11538

Irrespective of bank, compilation of year wise rate of interest and EMI are as under:

Years	Rate	EMI
5	8.75	20637
5	9.00	20758
5	9.25	20880
5	9.50	21002
5	9.75	21124
5	10.00	21247
5	10.25	21370
5	10.50	21494
5	10.75	21618
5	11.00	21742
5	11.25	21867
5	11.75	22118
5	12.00	22244
5	12.50	22498
10	8.75	12533
10	9.25	12803
10	9.50	12940
10	9.75	13077
10	10.00	13215
10	10.25	13354
10	10.75	13634
10	11.00	13775
10	11.75	14203
10	12.50	14638
10	13.50	15227
15	9.25	10292
15	10.00	10746
15	10.25	10900
15	10.50	11054
15	11.00	11366
15	12.00	12002
15	12.75	12488
15	13.00	12652
20	9.25	9159
20	10.25	9816
20	10.50	9984
20	11.00	10322
20	12.00	11011
20	12.75	11538

4.3 PERSONAL LOAN

4.3.1 PROGRAM TO FIND EMI ON REDUCING MONTHLY BASIS

We have prepared a C++ program 'emiperm.cpp' to generate EMIs on reducing monthly basis for the term of 3 years and interest rate ranging from 10.00 to 18.00 after every intervals of 0.25, i.e. 10.00, 10.25, 10.50,…..17.50, 17.75, 18.00 for the amount of Rs.100000.00. The output of this program is shown at figure 4.4.

FIGURE 4.4 Personal loan EMIs for Rs.1,00,000.00 (reducing monthly basis)

Rate of Interest	3 Yrs EMI	Total Interest	Total to pay
10.00	3227	16161.88	116161.88
10.25	3238	16584.88	116584.88
10.50	3250	17008.80	117008.80
10.75	3262	17433.63	117433.63
11.00	3274	17859.38	117859.38
11.25	3286	18286.05	118286.05
11.50	3298	18713.62	118713.62
11.75	3310	19142.12	119142.12
12.00	3321	19571.52	119571.52
12.25	3333	20001.83	120001.83
12.50	3345	20433.05	120433.05
12.75	3357	20865.19	120865.19
13.00	3369	21298.23	121298.23
13.25	3381	21732.18	121732.18
13.50	3394	22167.04	122167.04
13.75	3406	22602.80	122602.80
14.00	3418	23039.47	123039.47
14.25	3430	23477.04	123477.04
14.50	3442	23915.52	123915.52
14.75	3454	24354.90	124354.90
15.00	3467	24795.19	124795.19
15.25	3479	25236.38	125236.38
15.50	3491	25678.45	125678.45
15.75	3503	26121.44	126121.44
16.00	3516	26565.32	126565.32
16.25	3528	27010.10	127010.10
16.50	3540	27455.77	127455.77
16.75	3553	27902.35	127902.35
17.00	3565	28349.82	128349.82
17.25	3578	28798.18	128798.18
17.50	3590	29247.44	129247.44
17.75	3603	29697.59	129697.59
18.00	3615	30148.62	130148.62

4.3.2 PROGRAM TO FIND EMI ON REDUCING DAILY BASIS

We have prepared a C++ program 'emiperd.cpp' to generate EMIs on reducing daily basis for the term of 3 years and interest rate ranging from 10.00 to 18.00 after every intervals of 0.25, i.e. 10.00, 10.25, 10.50,…..17.50, 17.75, 18.00 for the amount of Rs.100000.00. The output of this program is shown at figure 4.5.

FIGURE 4.5 Personal loan EMIs for Rs.1,00,000.00 (reducing daily basis)

Rate of Interest	3 Yrs EMI	Total Interest	Total to pay
10.00	3216	15762.48	115762.48
10.25	3227	16175.66	116175.66
10.50	3239	16589.77	116589.77
10.75	3250	17004.81	117004.81
11.00	3262	17420.78	117420.78
11.25	3273	17837.69	117837.69
11.50	3285	18255.51	118255.51
11.75	3297	18674.27	118674.27
12.00	3308	19093.93	119093.93
12.25	3320	19514.53	119514.53
12.50	3332	19936.06	119936.06
12.75	3343	20358.52	120358.52
13.00	3355	20781.88	120781.88
13.25	3367	21206.18	121206.18
13.50	3379	21631.40	121631.40
13.75	3390	22057.53	122057.53
14.00	3402	22484.59	122484.59
14.25	3414	22912.58	122912.58
14.50	3426	23341.48	123341.48
14.75	3438	23771.30	123771.30
15.00	3450	24202.03	124202.03
15.25	3462	24633.70	124633.70
15.50	3474	25066.27	125066.27
15.75	3486	25499.77	125499.77
16.00	3498	25934.17	125934.17
16.25	3510	26369.49	126369.49
16.50	3522	26805.73	126805.73
16.75	3535	27242.88	127242.88
17.00	3547	27680.95	127680.95
17.25	3559	28119.93	128119.93
17.50	3571	28559.82	128559.82
17.75	3583	29000.62	129000.62
18.00	3596	29442.34	129442.34

4.3.3 PROGRAM TO FIND THE DIFFERENCE OF EMI ON REDUCING MONTHLY AND DAILY BASIS

To see the clear difference of EMI amount of personal loan for daily and monthly reducing basis, prepared a C++ program ‘emidiffp.cpp’. The output of this program shows the difference of EMI for various rates of interest and terms as shown in figure 4.6.

FIGURE 4.6

Personal loan of Rs.100000/- for 3 years
EMIs on reducig monthly basis, daily basis and difference

Rate of Interest	Amount of EMI		
	monthly	daily	difference
10.00	3226.72	3215.62	399.40
10.25	3238.47	3227.10	409.21
10.50	3250.24	3238.60	419.03
10.75	3262.05	3250.13	428.82
11.00	3273.87	3261.69	438.59
11.25	3285.72	3273.27	448.37
11.50	3297.60	3284.88	458.11
11.75	3309.50	3296.51	467.85
12.00	3321.43	3308.16	477.58
12.25	3333.38	3319.85	487.29
12.50	3345.36	3331.56	496.99
12.75	3357.37	3343.29	506.67
13.00	3369.40	3355.05	516.35
13.25	3381.45	3366.84	526.00
13.50	3393.53	3378.65	535.64
13.75	3405.63	3390.49	545.26
14.00	3417.76	3402.35	554.87
14.25	3429.92	3414.24	564.46
14.50	3442.10	3426.15	574.04
14.75	3454.30	3438.09	583.60
15.00	3466.53	3450.06	593.16
15.25	3478.79	3462.05	602.67
15.50	3491.07	3474.06	612.18
15.75	3503.37	3486.10	621.68
16.00	3515.70	3498.17	631.15
16.25	3528.06	3510.26	640.61
16.50	3540.44	3522.38	650.05
16.75	3552.84	3534.52	659.47
17.00	3565.27	3546.69	668.87
17.25	3577.73	3558.89	678.25
17.50	3590.21	3571.11	687.61
17.75	3602.71	3583.35	696.96
18.00	3615.24	3595.62	706.29

4.3.4 RATE OF INTEREST AND EMI OF VARIOUS BANKS FOR DIFFERENT PERIODS

For the academic interest, we have collected personal loan interest rates and EMI amount of various banks from the web site ecompare.co.in. As our interest is only to know the amount of EMI at certain rate of interest and certain tenure, the lender bank and the effective date is not important for us. Table 4.6 shows interest rates and EMI of various banks for the personal loan of Rs.1,00,000 over tenure of 3 years. That will be useful to compare our program generated EMIs:

TABLE 4.2 Personal Loan Search Results for 1,00,000 over 3 years

Updated on 01 March 2010

Sr. No.	Financial Institution	Interest Rate	EMI	Total Interest	Total To Pay
1	Bank of India	11.50%	3298	18714	118714
2	Union Bank of India	11.75%	3310	19142	119142
3	United Bank	12.75%	3357	20865	120865
4	Central Bank of India	13.00%	3369	21298	121298
5	Allahabad Bank	13.00%	3369	21298	121298
6	Oriental Bank	13.00%	3369	21298	121298
7	Vijaya Bank	13.25%	3381	21732	121732
8	UCO Bank	13.25%	3381	21732	121732
9	Canara Bank	13.75%	3406	22603	122603
10	Indian Bank	13.75%	3406	22603	122603
11	Punjab & Sind Bank	14.00%	3418	23039	123039
12	Bank of Maharashtra	14.25%	3430	23477	123477
13	Corporation Bank	14.50%	3442	23916	123916
14	Dena Bank	15.00%	3467	24795	124795
15	South Indian Bank	15.25%	3479	25236	125236
16	Bank of Baroda	15.25%	3479	25236	125236
17	Karur Vysya Bank	15.50%	3491	25678	125678
18	State Bank of India	16.00%	3516	26565	126565
19	Federal Bank	16.25%	3528	27010	127010
20	IndusInd Bank	17.00%	3565	28350	128350
21	ABN-AMRO	17.75%	3603	29698	129698
22	Kotak	19.00%	3666	31962	131962

4.4 CAR LOAN

4.4.1 PROGRAM TO FIND EMI ON REDUCING MONTHLY BASIS

We have prepared a C++ program 'emicarm.cpp' to generate EMIs on reducing monthly basis for the term of 3 years and interest rate ranging from 8.00 to 15.00 after every intervals of 0.25, i.e. 8.00, 8.25, 8.50,…..14.50, 14.75, 15.00 for the amount of Rs.100000.00. The output of this program is shown at figure 4.7.

FIGURE 4.7 Car loan EMIs for Rs.1,00,000.00 (reducing monthly basis)

Rate of Interest	3 Yrs EMI	Total Interest	Total to pay
8.00	3134	12810.91	112810.91
8.25	3145	13226.56	113226.56
8.50	3157	13643.13	113643.13
8.75	3168	14060.63	114060.63
9.00	3180	14479.03	114479.03
9.25	3192	14898.37	114898.37
9.50	3203	15318.62	115318.62
9.75	3215	15739.79	115739.79
10.00	3227	16161.88	116161.88
10.25	3238	16584.88	116584.88
10.50	3250	17008.80	117008.80
10.75	3262	17433.63	117433.63
11.00	3274	17859.38	117859.38
11.25	3286	18286.05	118286.05
11.50	3298	18713.62	118713.62
11.75	3310	19142.12	119142.12
12.00	3321	19571.52	119571.52
12.25	3333	20001.83	120001.83
12.50	3345	20433.05	120433.05
12.75	3357	20865.19	120865.19
13.00	3369	21298.23	121298.23
13.25	3381	21732.18	121732.18
13.50	3394	22167.04	122167.04
13.75	3406	22602.80	122602.80
14.00	3418	23039.47	123039.47
14.25	3430	23477.04	123477.04
14.50	3442	23915.52	123915.52
14.75	3454	24354.90	124354.90
15.00	3467	24795.19	124795.19

4.4.2 PROGRAM TO FIND EMI ON REDUCING DAILY BASIS

We have prepared a C++ program 'emicard.cpp' to generate EMIs on reducing monthly basis for the term of 3 years and interest rate ranging from 8.00 to 15.00 after every intervals of 0.25, i.e. 8.00, 8.25, 8.50,…..14.50, 14.75, 15.00 for the amount of Rs.1,00,000.00. The output of this program is shown at figure 4.8.

FIGURE 4.8 Car loan EMIs for Rs.1,00,000.00 (reducing daily basis)

Rate of Interest	3 Yrs EMI	Total Interest	Total to pay
8.00	3125	12490.45	112490.45
8.25	3136	12896.20	112896.20
8.50	3147	13302.88	113302.88
8.75	3159	13710.48	113710.48
9.00	3170	14119.02	114119.02
9.25	3181	14528.49	114528.49
9.50	3193	14938.88	114938.88
9.75	3204	15350.21	115350.21
10.00	3216	15762.48	115762.48
10.25	3227	16175.66	116175.66
10.50	3239	16589.77	116589.77
10.75	3250	17004.81	117004.81
11.00	3262	17420.78	117420.78
11.25	3273	17837.69	117837.69
11.50	3285	18255.51	118255.51
11.75	3297	18674.27	118674.27
12.00	3308	19093.93	119093.93
12.25	3320	19514.53	119514.53
12.50	3332	19936.06	119936.06
12.75	3343	20358.52	120358.52
13.00	3355	20781.88	120781.88
13.25	3367	21206.18	121206.18
13.50	3379	21631.40	121631.40
13.75	3390	22057.53	122057.53
14.00	3402	22484.59	122484.59
14.25	3414	22912.58	122912.58
14.50	3426	23341.48	123341.48
14.75	3438	23771.30	123771.30
15.00	3450	24202.03	124202.03

4.4.3 PROGRAM TO FIND THE DIFFERENCE OF EMI ON REDUCING MONTHLY AND DAILY BASIS

To see the clear difference of EMI amount of car loan for daily and monthly reducing basis, prepared a C++ program ‘emidiffc.cpp’. The output of this program shows the difference of EMI for various rates of interest and terms given at figure 4.9.

FIGURE 4.9

Car loan of Rs.100000/- for 3 years
EMIs on reducig monthly basis, daily basis and difference

Rate of Interest	Amount of EMI		
	monthly	daily	difference
8.00	3133.64	3124.73	320.47
8.25	3145.18	3136.01	330.37
8.50	3156.75	3147.30	340.26
8.75	3168.35	3158.62	350.16
9.00	3179.97	3169.97	360.02
9.25	3191.62	3181.35	369.88
9.50	3203.29	3192.75	379.73
9.75	3214.99	3204.17	389.58
10.00	3226.72	3215.62	399.40
10.25	3238.47	3227.10	409.21
10.50	3250.24	3238.60	419.03
10.75	3262.05	3250.13	428.82
11.00	3273.87	3261.69	438.59
11.25	3285.72	3273.27	448.37
11.50	3297.60	3284.88	458.11
11.75	3309.50	3296.51	467.85
12.00	3321.43	3308.16	477.58
12.25	3333.38	3319.85	487.29
12.50	3345.36	3331.56	496.99
12.75	3357.37	3343.29	506.67
13.00	3369.40	3355.05	516.35
13.25	3381.45	3366.84	526.00
13.50	3393.53	3378.65	535.64
13.75	3405.63	3390.49	545.26
14.00	3417.76	3402.35	554.87
14.25	3429.92	3414.24	564.46
14.50	3442.10	3426.15	574.04
14.75	3454.30	3438.09	583.60
15.00	3466.53	3450.06	593.16

4.4.4 RATE OF INTEREST AND EMI OF VARIOUS BANKS FOR DIFFERENT PERIODS

For the academic interest, we have collected car loan interest rates and EMI amount of various banks from the web site ecompare.co.in. As our interest is only to know the amount of EMI at certain rate of interest and certain tenure, the lender bank and the effective date is not important for us. Following tables show interest rates and EMI of various banks for the personal loan of Rs.100000 over tenure of 3 years. That will be useful to compare our program generated EMIs:

TABLE 4.3 Car Loan Search Results for Rs. 100,000 over 3 years

Updated on 01 March 2010

Sr. No.	Financial Institution	Interest Rate	EMI	Total Interest	Total To Pay
1	Bank of India	9.75%	3215	15740	115740
2	Union Bank of India	9.75%	3215	15740	115740
3	Central Bank Of India	10.00%	3227	16162	116162
4	Bank of Baroda	10.00%	3227	16162	116162
5	Bank of Maharashtra	10.25%	3238	16585	116585
6	Punjab National Bank	10.50%	3250	17009	117009
7	State Bank of India	10.50%	3250	17009	117009
8	Punjab & Sindh Bank	10.75%	3262	17434	117434
9	Indian Bank	10.75%	3262	17434	117434
10	United Bank	10.75%	3262	17434	117434
11	Dena Bank	11.00%	3274	17859	117859
12	Oriental Bank	11.00%	3274	17859	117859
13	Allahabad Bank	11.00%	3274	17859	117859
14	Vijaya Bank	11.25%	3286	18286	118286
15	Indian Overseas Bank	11.25%	3286	18286	118286
16	Canara Bank	11.25%	3286	18286	118286
17	Corporation Bank	11.50%	3250	17009	117009
18	Karur Vysya Bank	11.50%	3298	18714	118714
19	UCO Bank	11.50%	3298	18714	118714
20	Federal Bank	14.25%	3430	23477	123477
21	South Indian Bank	14.75%	3454	24355	124355
22	ABN-AMRO	16.75%	3553	27902	127902

4.5 VERIFICATION OF THE RESULT

As per program generated table, sample details of EMI for the housing loan at 7.5% rate of interest and loan amount of Rs.10,00,000.00 are as under:

5 years		10 years		15 years		20 years	
Monthly basis	Daily basis	Monthly basis	Daily basis	Monthly basis	Daily basis	Monthly basis	Daily basis
20038	19988	11870	11846	9270	9255	8056	8045

While taking any loan from the bank, we are paying EMI on monthly basis for opted number of months. Here interest is calculated on a ‘reducing balance’ or only on the

amount of loan left to pay and not the entire loan amount. So the loan left to pay will be reduced to zero after paying all instalments.

We have calculated opening balance, adjusted principal, adjusted interest and closing balance after every month using Excel as well as C++. We have prepared worksheet 'emi-check-final.xls' and also prepared a C++ program 'emicheck.cpp' for the required calculations of closing balance after every month of payment for the entire duration.

Following is the Excel worksheet based on EMI Rs. 20038.00, calculated on monthly basis @ 7.5% for 5 years. Brief illustration of each column is as under:

Here in following example we are repaying equal monthly instalments – EMI Rs. 20038.00 at 7.5% rate of interest every month for 60 months.

Date of depositing EMI varies, so at an average if we consider that customer is depositing the EMI in the middle of the month, then customer gets interest of 15 days for particular month. So the effective EMI is amount of EMI (EMI+ interest of 15 days on that).

Opening balance is outstanding loan amount at the beginning of the month. So the interest is calculated on opening balance for a particular month and that is adjusted from the paid EMI amount. The remaining amount (EMI-Adjusted interest) is adjusted principal. Now adjusted principal is subtracted from opening balance and this becomes closing balance for a particular month. This closing balance will become opening balance for the next month.

Thus for the proper amount of EMI, after repaying all the EMIs, closing balance should be zero.

TABLE 4.4 Calculations on Monthly Basis (Using EXCEL)

Sr	Rate of Interest	EMI	Amount of EMI(with Interest)	Opening Balance	Adjusted Principle	Adjusted Interest	Closing Balance
1	7.50	20037.95	20100.66	1000000.00	13831.73	6268.93	986168.27
2	7.50	20037.95	20100.66	986168.27	13918.44	6182.22	972249.82
3	7.50	20037.95	20100.66	972249.82	14005.70	6094.96	958244.12
4	7.50	20037.95	20100.66	958244.12	14093.50	6007.16	944150.63
5	7.50	20037.95	20100.66	944150.63	14181.85	5918.81	929968.78
6	7.50	20037.95	20100.66	929968.78	14270.75	5829.91	915698.02
7	7.50	20037.95	20100.66	915698.02	14360.22	5740.44	901337.80
8	7.50	20037.95	20100.66	901337.80	14450.24	5650.42	886887.56
9	7.50	20037.95	20100.66	886887.56	14540.83	5559.83	872346.74
10	7.50	20037.95	20100.66	872346.74	14631.98	5468.68	857714.75
11	7.50	20037.95	20100.66	857714.75	14723.71	5376.95	842991.04
12	7.50	20037.95	20100.66	842991.04	14816.01	5284.65	828175.03
13	7.50	20037.95	20100.66	828175.03	14908.89	5191.77	813266.14
14	7.50	20037.95	20100.66	813266.14	15002.35	5098.31	798263.78
15	7.50	20037.95	20100.66	798263.78	15096.40	5004.26	783167.38
16	7.50	20037.95	20100.66	783167.38	15191.04	4909.62	767976.34
17	7.50	20037.95	20100.66	767976.34	15286.27	4814.39	752690.07
18	7.50	20037.95	20100.66	752690.07	15382.10	4718.56	737307.96
19	7.50	20037.95	20100.66	737307.96	15478.53	4622.13	721829.43
20	7.50	20037.95	20100.66	721829.43	15575.56	4525.10	706253.87
21	7.50	20037.95	20100.66	706253.87	15673.21	4427.45	690580.66
22	7.50	20037.95	20100.66	690580.66	15771.46	4329.20	674809.20
23	7.50	20037.95	20100.66	674809.20	15870.33	4230.33	658938.87
24	7.50	20037.95	20100.66	658938.87	15969.82	4130.84	642969.05
25	7.50	20037.95	20100.66	642969.05	16069.93	4030.73	626899.11
26	7.50	20037.95	20100.66	626899.11	16170.68	3929.98	610728.44
27	7.50	20037.95	20100.66	610728.44	16272.05	3828.61	594456.39
28	7.50	20037.95	20100.66	594456.39	16374.06	3726.60	578082.33
29	7.50	20037.95	20100.66	578082.33	16476.70	3623.96	561605.63
30	7.50	20037.95	20100.66	561605.63	16580.00	3520.66	545025.63
31	7.50	20037.95	20100.66	545025.63	16683.93	3416.73	528341.70
32	7.50	20037.95	20100.66	528341.70	16788.53	3312.13	511553.17
33	7.50	20037.95	20100.66	511553.17	16893.77	3206.89	494659.40
34	7.50	20037.95	20100.66	494659.40	16999.68	3100.98	477659.72
35	7.50	20037.95	20100.66	477659.72	17106.25	2994.41	460553.48
36	7.50	20037.95	20100.66	460553.48	17213.48	2887.18	443339.99
37	7.50	20037.95	20100.66	443339.99	17321.39	2779.27	426018.60
38	7.50	20037.95	20100.66	426018.60	17429.98	2670.68	408588.62
39	7.50	20037.95	20100.66	408588.62	17539.25	2561.41	391049.37
40	7.50	20037.95	20100.66	391049.37	17649.20	2451.46	373400.17
41	7.50	20037.95	20100.66	373400.17	17759.84	2340.82	355640.32

42	7.50	20037.95	20100.66	355640.32	17871.18	2229.48	337769.15
43	7.50	20037.95	20100.66	337769.15	17983.21	2117.45	319785.94
44	7.50	20037.95	20100.66	319785.94	18095.95	2004.71	301689.99
45	7.50	20037.95	20100.66	301689.99	18209.39	1891.27	283480.60
46	7.50	20037.95	20100.66	283480.60	18323.54	1777.12	265157.06
47	7.50	20037.95	20100.66	265157.06	18438.41	1662.25	246718.65
48	7.50	20037.95	20100.66	246718.65	18554.00	1546.66	228164.65
49	7.50	20037.95	20100.66	228164.65	18670.31	1430.35	209494.34
50	7.50	20037.95	20100.66	209494.34	18787.36	1313.30	190706.99
51	7.50	20037.95	20100.66	190706.99	18905.13	1195.53	171801.85
52	7.50	20037.95	20100.66	171801.85	19023.65	1077.01	152778.21
53	7.50	20037.95	20100.66	152778.21	19142.90	957.76	133635.30
54	7.50	20037.95	20100.66	133635.30	19262.91	837.75	114372.39
55	7.50	20037.95	20100.66	114372.39	19383.67	716.99	94988.72
56	7.50	20037.95	20100.66	94988.72	19505.18	595.48	75483.54
57	7.50	20037.95	20100.66	75483.54	19627.46	473.20	55856.08
58	7.50	20037.95	20100.66	55856.08	19750.50	350.16	36105.58
59	7.50	20037.95	20100.66	36105.58	19874.32	226.34	16231.26
60	7.50	20037.95	20100.66	16231.26	19998.91	101.75	-3767.65

FIGURE 4.10 Calculations of closing balance on Monthly Basis (Using C++)

```
          Loan amount:1000000.00
      Rate of Interest:       7.50
 EMI on Monthly basis:   20037.95
 EMI+15 days interest:   20100.66
-------------------------------------------------------------
Month       Opening     Adjusted     Adjusted      Closing
             Balance    Principal     Interest      Balance
-------------------------------------------------------------
    1   1000000.00     13831.73      6268.93    986168.27
    2    986168.27     13918.44      6182.22    972249.82
    3    972249.82     14005.70      6094.96    958244.12
    4    958244.12     14093.50      6007.16    944150.63
    5    944150.63     14181.85      5918.81    929968.78
    6    929968.78     14270.75      5829.91    915698.02
    7    915698.02     14360.22      5740.44    901337.80
    8    901337.80     14450.24      5650.42    886887.56
    9    886887.56     14540.83      5559.83    872346.74
   10    872346.74     14631.98      5468.68    857714.75
   11    857714.75     14723.71      5376.95    842991.04
   12    842991.04     14816.01      5284.65    828175.03
   13    828175.03     14908.89      5191.77    813266.14
   14    813266.14     15002.35      5098.31    798263.78
   15    798263.78     15096.40      5004.26    783167.38
   16    783167.38     15191.04      4909.62    767976.34
   17    767976.34     15286.27      4814.39    752690.07
   18    752690.07     15382.10      4718.56    737307.96
   19    737307.96     15478.53      4622.13    721829.43
   20    721829.43     15575.56      4525.10    706253.87
   21    706253.87     15673.21      4427.45    690580.66
   22    690580.66     15771.46      4329.20    674809.20
   23    674809.20     15870.33      4230.33    658938.87
   24    658938.87     15969.82      4130.84    642969.05
   25    642969.05     16069.93      4030.73    626899.11
   26    626899.11     16170.68      3929.98    610728.44
   27    610728.44     16272.05      3828.61    594456.39
   28    594456.39     16374.06      3726.60    578082.33
   29    578082.33     16476.70      3623.96    561605.63
   30    561605.63     16580.00      3520.66    545025.63
```

31	545025.63	16683.93	3416.73	528341.70
32	528341.70	16788.53	3312.13	511553.17
33	511553.17	16893.77	3206.89	494659.40
34	494659.40	16999.68	3100.98	477659.72
35	477659.72	17106.25	2994.41	460553.48
36	460553.48	17213.48	2887.18	443339.99
37	443339.99	17321.39	2779.27	426018.60
38	426018.60	17429.98	2670.68	408588.62
39	408588.62	17539.25	2561.41	391049.37
40	391049.37	17649.20	2451.46	373400.17
41	373400.17	17759.84	2340.82	355640.32
42	355640.32	17871.18	2229.48	337769.15
43	337769.15	17983.21	2117.45	319785.94
44	319785.94	18095.95	2004.71	301689.99
45	301689.99	18209.39	1891.27	283480.60
46	283480.60	18323.54	1777.12	265157.06
47	265157.06	18438.41	1662.25	246718.65
48	246718.65	18554.00	1546.66	228164.65
49	228164.65	18670.31	1430.35	209494.34
50	209494.34	18787.36	1313.30	190706.99
51	190706.99	18905.13	1195.53	171801.85
52	171801.85	19023.65	1077.01	152778.21
53	152778.21	19142.90	957.76	133635.30
54	133635.30	19262.91	837.75	114372.39
55	114372.39	19383.67	716.99	94988.72
56	94988.72	19505.18	595.48	75483.54
57	75483.54	19627.46	473.20	55856.08
58	55856.08	19750.50	350.16	36105.58
59	36105.58	19874.32	226.34	16231.26
60	16231.26	19998.91	101.75	-3767.65

From the table 4.8 and figure 4.10, it is clear that customer is repaying Rs.3767.65 more for this case when EMI is calculated on monthly compounding basis.

Following is the Excel worksheet based on EMI Rs. 19988.00, calculated on daily basis for 5 years.

TABLE 4.5 Calculations on Daily Basis (Using EXCEL)

Sr	Rate of Interest	EMI	Amount of EMI(with Interest)	Opening Balance	Adjusted Principle	Adjusted Interest	Closing Balance
1	7.50	19988.22	20050.77	1000000.00	13781.85	6268.93	986218.15
2	7.50	19988.22	20050.77	986218.15	13868.25	6182.53	972349.91
3	7.50	19988.22	20050.77	972349.91	13955.18	6095.59	958394.72
4	7.50	19988.22	20050.77	958394.72	14042.67	6008.11	944352.05
5	7.50	19988.22	20050.77	944352.05	14130.70	5920.07	930221.35
6	7.50	19988.22	20050.77	930221.35	14219.29	5831.49	916002.06
7	7.50	19988.22	20050.77	916002.06	14308.43	5742.35	901693.64
8	7.50	19988.22	20050.77	901693.64	14398.12	5652.65	887295.52
9	7.50	19988.22	20050.77	887295.52	14488.38	5562.39	872807.13
10	7.50	19988.22	20050.77	872807.13	14579.21	5471.56	858227.92
11	7.50	19988.22	20050.77	858227.92	14670.61	5380.17	843557.31
12	7.50	19988.22	20050.77	843557.31	14762.58	5288.20	828794.74
13	7.50	19988.22	20050.77	828794.74	14855.12	5195.65	813939.61
14	7.50	19988.22	20050.77	813939.61	14948.25	5102.53	798991.37
15	7.50	19988.22	20050.77	798991.37	15041.96	5008.82	783949.41
16	7.50	19988.22	20050.77	783949.41	15136.25	4914.52	768813.16

17	7.50	19988.22	20050.77	768813.16	15231.14	4819.63	753582.02
18	7.50	19988.22	20050.77	753582.02	15326.62	4724.15	738255.39
19	7.50	19988.22	20050.77	738255.39	15422.71	4628.07	722832.68
20	7.50	19988.22	20050.77	722832.68	15519.39	4531.38	707313.29
21	7.50	19988.22	20050.77	707313.29	15616.68	4434.09	691696.62
22	7.50	19988.22	20050.77	691696.62	15714.58	4336.19	675982.04
23	7.50	19988.22	20050.77	675982.04	15813.09	4237.68	660168.94
24	7.50	19988.22	20050.77	660168.94	15912.22	4138.55	644256.72
25	7.50	19988.22	20050.77	644256.72	16011.98	4038.80	628244.74
26	7.50	19988.22	20050.77	628244.74	16112.35	3938.42	612132.39
27	7.50	19988.22	20050.77	612132.39	16213.36	3837.41	595919.02
28	7.50	19988.22	20050.77	595919.02	16315.00	3735.77	579604.02
29	7.50	19988.22	20050.77	579604.02	16417.28	3633.49	563186.74
30	7.50	19988.22	20050.77	563186.74	16520.20	3530.58	546666.54
31	7.50	19988.22	20050.77	546666.54	16623.76	3427.01	530042.78
32	7.50	19988.22	20050.77	530042.78	16727.98	3322.80	513314.81
33	7.50	19988.22	20050.77	513314.81	16832.84	3217.93	496481.96
34	7.50	19988.22	20050.77	496481.96	16938.37	3112.41	479543.60
35	7.50	19988.22	20050.77	479543.60	17044.55	3006.22	462499.05
36	7.50	19988.22	20050.77	462499.05	17151.40	2899.37	445347.65
37	7.50	19988.22	20050.77	445347.65	17258.92	2791.85	428088.72
38	7.50	19988.22	20050.77	428088.72	17367.12	2683.66	410721.60
39	7.50	19988.22	20050.77	410721.60	17475.99	2574.78	393245.61
40	7.50	19988.22	20050.77	393245.61	17585.55	2465.23	375660.07
41	7.50	19988.22	20050.77	375660.07	17695.79	2354.99	357964.28
42	7.50	19988.22	20050.77	357964.28	17806.72	2244.05	340157.55
43	7.50	19988.22	20050.77	340157.55	17918.35	2132.42	322239.20
44	7.50	19988.22	20050.77	322239.20	18030.68	2020.09	304208.52
45	7.50	19988.22	20050.77	304208.52	18143.71	1907.06	286064.81
46	7.50	19988.22	20050.77	286064.81	18257.46	1793.32	267807.35
47	7.50	19988.22	20050.77	267807.35	18371.91	1678.86	249435.44
48	7.50	19988.22	20050.77	249435.44	18487.08	1563.69	230948.36
49	7.50	19988.22	20050.77	230948.36	18602.98	1447.80	212345.38
50	7.50	19988.22	20050.77	212345.38	18719.60	1331.18	193625.79
51	7.50	19988.22	20050.77	193625.79	18836.95	1213.83	174788.84
52	7.50	19988.22	20050.77	174788.84	18955.04	1095.74	155833.80
53	7.50	19988.22	20050.77	155833.80	19073.86	976.91	136759.94
54	7.50	19988.22	20050.77	136759.94	19193.44	857.34	117566.50
55	7.50	19988.22	20050.77	117566.50	19313.76	737.02	98252.74
56	7.50	19988.22	20050.77	98252.74	19434.84	615.94	78817.91
57	7.50	19988.22	20050.77	78817.91	19556.67	494.10	59261.24
58	7.50	19988.22	20050.77	59261.24	19679.27	371.50	39581.97
59	7.50	19988.22	20050.77	39581.97	19802.64	248.14	19779.33
60	7.50	19988.22	20050.77	19779.33	19926.78	124.00	-147.45

FIGURE 4.11 Calculations of closing balance on Daily Basis (Using C++)

Loan amount:1000000.00
Rate of Interest: 7.50
EMI on Monthly basis: 19988.22
EMI+15 days interest: 20050.77

Month	Opening Balance	Adjusted Principal	Adjusted Interest	Closing Balance
1	1000000.00	13781.85	6268.93	986218.15
2	986218.15	13868.25	6182.53	972349.91
3	972349.91	13955.18	6095.59	958394.72
4	958394.72	14042.67	6008.11	944352.05
5	944352.05	14130.70	5920.07	930221.35
6	930221.35	14219.29	5831.49	916002.06
7	916002.06	14308.43	5742.35	901693.64
8	901693.64	14398.12	5652.65	887295.52
9	887295.52	14488.38	5562.39	872807.13
10	872807.13	14579.21	5471.56	858227.92
11	858227.92	14670.61	5380.17	843557.31
12	843557.31	14762.58	5288.20	828794.74
13	828794.74	14855.12	5195.65	813939.61
14	813939.61	14948.25	5102.53	798991.37
15	798991.37	15041.96	5008.82	783949.41
16	783949.41	15136.25	4914.52	768813.16
17	768813.16	15231.14	4819.63	753582.02
18	753582.02	15326.62	4724.15	738255.39
19	738255.39	15422.71	4628.07	722832.68
20	722832.68	15519.39	4531.38	707313.29
21	707313.29	15616.68	4434.09	691696.62
22	691696.62	15714.58	4336.19	675982.04
23	675982.04	15813.09	4237.68	660168.94
24	660168.94	15912.22	4138.55	644256.72
25	644256.72	16011.98	4038.80	628244.74
26	628244.74	16112.35	3938.42	612132.39
27	612132.39	16213.36	3837.41	595919.02
28	595919.02	16315.00	3735.77	579604.02
29	579604.02	16417.28	3633.49	563186.74
30	563186.74	16520.20	3530.58	546666.54
31	546666.54	16623.76	3427.01	530042.78
32	530042.78	16727.98	3322.80	513314.81
33	513314.81	16832.84	3217.93	496481.96
34	496481.96	16938.37	3112.41	479543.60
35	479543.60	17044.55	3006.22	462499.05
36	462499.05	17151.40	2899.37	445347.65
37	445347.65	17258.92	2791.85	428088.72
38	428088.72	17367.12	2683.66	410721.60
39	410721.60	17475.99	2574.78	393245.61
40	393245.61	17585.55	2465.23	375660.07
41	375660.07	17695.79	2354.99	357964.28
42	357964.28	17806.72	2244.05	340157.55
43	340157.55	17918.35	2132.42	322239.20
44	322239.20	18030.68	2020.09	304208.52
45	304208.52	18143.71	1907.06	286064.81
46	286064.81	18257.46	1793.32	267807.35
47	267807.35	18371.91	1678.86	249435.44
48	249435.44	18487.08	1563.69	230948.36
49	230948.36	18602.98	1447.80	212345.38
50	212345.38	18719.60	1331.18	193625.79
51	193625.79	18836.95	1213.83	174788.84
52	174788.84	18955.04	1095.74	155833.80
53	155833.80	19073.86	976.91	136759.94
54	136759.94	19193.44	857.34	117566.50
55	117566.50	19313.76	737.02	98252.74
56	98252.74	19434.84	615.94	78817.91
57	78817.91	19556.67	494.10	59261.24
58	59261.24	19679.27	371.50	39581.97
59	39581.97	19802.64	248.14	19779.33
60	19779.33	19926.78	124.00	-147.45

From the table 4.9 and figure 4.11, it is clear that customer is repaying only Rs.147.45 more for this case when EMI is calculated on daily compounding basis.

Following is the output of 'emichkal.cpp', showing Summary of closing balances for both the methods after completion of various durations is as under:

FIGURE 4.12

		5 years	10 years	15 years	20 years
Monthly	EMI	20037.95	11870.18	9270.12	8055.93
	Closing Balance	-3767.65	-4506.39	-5340.63	-6284.01
Daily	EMI	19988.22	11846.16	9255.01	8045.46
	Closing Balance	-147.45	-213.73	-311.85	-452.05

From above figure 4.12, we have prepared an Excel worksheet 'EMI check all.xls' given as under:

Rest Basis	EMI	Number of installments	Total Repayment	Closing Balance	
				Amount	Extra % on loan amount
Monthly	20037.95	60	1202277.00	-3767.65	0.38
	11870.18	120	1424421.60	-4506.39	0.45
	9270.12	180	1668621.60	-5340.63	0.53
	8055.93	240	1933423.20	-6284.01	0.63
Daily	19988.22	60	1199293.20	-147.45	0.01
	11846.16	120	1421539.20	-213.73	0.02
	9255.01	180	1665901.80	-311.85	0.03
	8045.46	240	1930910.40	-452.05	0.05

Here, in principle, after completion of all instalments the closing balance amount should be 0. But considering rounding errors and other points like 365 days in a year, the amount of closing balance is approaching to 0.

Thus EMI calculated based on daily basis is the proper one.

Chapter

5

Effect of Rest Basis on Bank Deposits

5.1 INTRODUCTION

Banks are also called custodians of public money. Basically, the money is accepted as deposit for safe keeping. Since the Banks use this money to earn interest from people who need money, Banks share a part of this interest with the depositors. The quantum of interest depends upon the term - length of time for which the depositor wishes to keep the money with the Bank so to make withdrawal.

A fixed deposit is meant for those investors who want to deposit a lump sum of money for a fixed period; say for a minimum period of 15 days to five years and above, thereby earning a higher rate of interest in return. Investor gets a lump sum (principal + interest) at the maturity of the deposit.

One can calculate maturity amount using different methods of compounding. Here in this chapter we have calculated maturity amounts of fixed deposits and recurring deposits using method of quarterly compounding and method of daily compounding. We have also found the difference amount by using both the methods.

5.1.1 FEATURES

Bank deposits are fairly safe because banks are subject to control of the Reserve Bank of India (RBI) with regard to several policy and operational parameters. The banks are free to offer varying interests in fixed deposits of different maturities. Interest is compounded once a quarter, leading to a somewhat higher effective rate.

5.1.2 RETURNS

The rate of interest for Bank Fixed Deposits varies between 4 and 11 per cent, depending on the maturity period (duration) of the FD and the amount invested. Interest rate also varies between each bank. A Bank FD does not provide regular interest income, but a lump-sum amount on its maturity. Some banks have facility to pay interest every quarter or every month, but the interest paid may be at a discounted rate in case of monthly interest. The Interest payable on Fixed Deposit can also be transferred to Savings Bank or Current Account of the customer. The deposit period can vary from 15, 30 or 45 days to 3, 6 months, 1 year, 1.5 years to 10 years.

Duration	**Interest rate (%) per annum**
15-30 days	4 -5 %
30-45 days	4.25-5 %
46-90 days	4.75—5.5 %
91-180 days	5.5-6.5 %
181-365 days	5.75-6.5 %
1-2 years	6-8 %
2-3 years	6.25-8 %
3-5 years	6.75-8

Indian Banks' Association (IBA) Code for Banking Practice is issued by IBA for uniform adoption by the Member Banks. The Code is intended to promote good banking practices by setting out minimum standards which Member Banks will follow in their dealings with customers. IBA, for the purpose of calculation of interest on domestic term deposit, have prescribed that on deposits repayable in less than three months or where the terminal quarter is incomplete, interest should be paid proportionately for the actual number of days reckoning the year at 365/ 366 days.

5.1.3 BENEFITS, DRAWBACKS AND PRECAUTIONS

Any investment portfolio should comprise the right of safe, moderate and risky investments. Mutual funds and stocks are the favorite contenders for moderate and risky investments. Fixed deposits, government bonds etc. are considered safe investments. Fixed deposits have been particularly popular among a large section of investors in India as a safe investment option for a long period.

With fixed deposits or FDs as they are popularly known, a person can invest an amount for a fixed duration. The banks provide interest rates depending on this loan amount and the term of deposit. Here are the benefits, drawbacks of fixed deposits and precautions one should take while making such investments.

5.1.3.1 BENEFITS

- **Safety:** The fixed deposits of reputed banks and financial institutions regulated by RBI (Reserve Bank of India) the banking regulator in India are very secure and considered as one of the safest investment methods.
- **Regular Income:** Fixed deposits earn fixed interest rates for their entire tenure, which is usually compounded quarterly. So, those who want an income on a regular basis can invest into fixed deposits and use the interest rate as their income. This makes a fixed deposit very popular way of investing money for retirees.
- **Saves tax:** With the directives of the income tax department stating that investment in fixed deposits up to a maximum of Rs.100,000 for 5 years are eligible for tax deductions under section 80 C of income tax act, fixed deposits have again become popular. Fixed deposits save tax and give high returns on invested money.

5.1.3.2 DRAWBACKS

- **Lower rate of returns:** While the money invested in stock markets may give you a return of 20% the fixed deposits will yield only about 7.5%. So, the money grows slowly in the case of fixed deposits.
- **Taxes:** The interest earned on fixed deposits is fully taxable and is added to the annual income of the individual. Gains from stocks are considered capital gains while dividends are tax free.
- **Rising inflation can wipe out the interest benefits:** The actual benefits or income from fixed deposit can be annulled by a rising inflation. Suppose the inflation which is currently at 3 % rises to about 6%, your fixed deposit at 7.5% annual return will effectively yield only(7.5%-6%) = 1.5% of return.

This return would have been (7.5% -3%) = 4.5% if the rate of inflation had not changed.

5.1.3.3 PRECAUTIONS

- **Company fixed deposits:** Company fixed deposits are not considered as safe as fixed deposits from leading banks and financial institutions regulated by the RBI. So, if a company runs into losses or goes bankrupt the money invested into its fixed deposit can be lost. To lure investors, such companies offer a fixed deposit interest rate which is much higher than those offered by banks. Before investing in any company fixed deposit scheme it is advised to check the credentials of the company.
- **Interest rate compounding period:** The interest rates offered on fixed deposit vary greatly with banks and tenures. Whether the interest rate is compounded annually, half yearly, quarterly or monthly will determine how much a person earns from his fixed deposit. A fixed deposit with interest rate compounded monthly will earn more than one which is compounded quarterly. It is therefore advised to shop around for the right fixed deposit scheme.
- **Premature ending of fixed deposits:** Banks will impose a penalty if you break your fixed deposit before the maturity period. Make sure you get the facts right about this thing. How the bank calculates this penalty and what all charges will it levy when you break a fixed deposit should be noted carefully.

5.2 METHODS OF COMPOUNDING

While calculating the compound interest, when the interest is compounded at different periodicity other than every year, the formula for compound interest calculation changes slightly. When interest is compounded

Yearly	The Principal changes Every year	Interest is calculated once a year(t=1)
Half yearly	The Principal changes Every half year	Interest is calculated twice a year(t=2)
Quarterly	The Principal changes Every quarter	Interest is calculated four times a year(t=4)

Monthly	The Principal changes Every month	Interest is calculated twelve times a year(t=12)
Daily	The Principal changes Every day	Interest is calculated 365 times a year(t=365)

Let R be the rate of interest per annum and N be the number of years for which the interest is calculated.

Then the formula for maturity amount changes to $A = P*(1+R/(t*100))^{N*t}$

The above change in formula is due to the fact that, the rate per year is converted to rate per half year(R/2), rate per quarter(R/4), rate per month(R/12), rate per day(R/365) if the interest is calculated half yearly(2 times), quarterly(4 times), monthly(12 times) or daily(365 times) respectively. Also, note that in such cases N changes to 2N, 4N, 12N and 365N respectively.

5.3 CALCULATION OF MATURITY AMOUNT

5.3.1 FIXED TERM DEPOSITS

To study the difference of Compound Interest on a Principal of Rs 1,00,000.00 @ different Rates (6% to 10%) for different terms (12, 15, 18, … , 51 months) we have applied both the methods. Like Compound Interest calculation on Quarterly compounding method and Compound Interest calculation on Daily compounding method.

While applying quarterly compounding method, we are compounding 4 times in a year, formula for maturity amount is: $Amount = P \times \left(1 + \frac{R}{400}\right)^{(4 \times N)}$

Example of this calculation: Let P=100000, R=6.00% and N=12 months,

$$Amount = 100000 \times \left(1 + \frac{6}{400}\right)^{(4 \times 12/12)} = 106136.35$$

We have written a C++ program ‘fdqtrly.cpp’ for the computation of maturity amount on quarterly compounded basis. The output for various interest rates is shown at figure 5.1.

FIGURE 5.1

Fixed Deposite of Rs.100000/-
Statement showing Amount returned on Quarterly compounded basis

	PERIOD IN MONTHS						
Rate	12	15	18	21	24	27	30
6.00	106136.35	107728.40	109344.33	110984.49	112649.26	114339.00	116054.09
6.25	106398.02	108060.48	109748.93	111463.76	113205.38	114974.21	116770.69
6.50	106660.16	108393.39	110154.78	111944.80	113763.90	115612.56	117491.27
6.75	106922.79	108727.11	110561.88	112427.62	114324.83	116254.06	118215.85
7.00	107185.91	109061.66	110970.23	112912.21	114888.18	116898.72	118944.45
7.25	107449.50	109397.02	111379.84	113398.60	115453.95	117546.55	119677.09
7.50	107713.59	109733.22	111790.71	113886.79	116022.17	118197.59	120413.79
7.75	107978.16	110070.23	112202.84	114376.77	116592.83	118851.81	121154.56
8.00	108243.22	110408.08	112616.24	114868.57	117165.94	119509.26	121899.45
8.25	108508.77	110746.76	113030.91	115362.17	117741.52	120169.94	122648.44
8.50	108774.80	111086.26	113446.84	115857.59	118319.56	120833.85	123401.57
8.75	109041.32	111426.60	113864.05	116354.83	118900.09	121501.03	124158.87
9.00	109308.34	111767.77	114282.55	116853.90	119483.12	122171.48	124920.34
9.25	109575.84	112109.77	114702.31	117354.80	120068.63	122845.23	125686.02
9.50	109843.83	112452.62	115123.37	117857.55	120656.66	123522.26	126455.91
9.75	110112.31	112796.30	115545.71	118362.13	121247.21	124202.62	127230.05
10.00	110381.29	113140.82	115969.34	118868.58	121840.29	124886.30	128008.45

Rate	33	36	39	42	45	48	51
6.00	117794.89	119561.82	121355.24	123175.57	125023.20	126898.55	128802.03
6.25	118595.23	120448.27	122330.28	124241.70	126182.97	128154.58	130156.99
6.50	119400.50	121340.76	123312.55	125316.38	127352.77	129422.25	131525.36
6.75	120210.74	122239.30	124302.09	126399.69	128532.68	130701.66	132907.27
7.00	121025.98	123143.93	125298.95	127491.68	129722.79	131992.94	134302.81
7.25	121846.23	124054.70	126303.19	128592.44	130923.17	133296.16	135712.14
7.50	122671.55	124971.64	127314.86	129702.01	132133.92	134611.44	137135.41
7.75	123501.94	125894.79	128334.00	130820.47	133355.11	135938.88	138572.69
8.00	124337.43	126824.18	129360.66	131947.88	134586.83	137278.56	140024.14
8.25	125178.06	127759.86	130394.91	133084.30	135829.17	138630.64	141489.91
8.50	126023.86	128701.87	131436.78	134229.81	137082.19	139995.19	142970.09
8.75	126874.84	129650.23	132486.33	135384.47	138346.00	141372.33	144464.84
9.00	127731.05	130605.00	133543.61	136548.34	139620.69	142762.14	145974.30
9.25	128592.51	131566.20	134608.67	137721.50	140906.31	144164.77	147498.58
9.50	129459.24	132533.91	135681.58	138904.02	142202.98	145580.31	149037.84
9.75	130331.28	133508.11	136762.38	140095.95	143510.80	147008.86	150592.20
10.00	131208.67	134488.89	137851.11	141297.39	144829.81	148450.56	152161.83

While applying daily compounding method, we are compounding 365 times in a year, formula for maturity amount is: $Amount = P \times \left(1 + \frac{R}{36500}\right)^{(365 \times N)}$,

Example of this calculation: Let P=100000, R=6.00% and N=12 months

$$Amount = 100000 \times \left(1 + \frac{6}{36500}\right)^{(365 \times 12/12)} = 106183.13$$

We have written a C++ program ‘fddaily.cpp’ for the computation of maturity amount on daily compounded basis. The output for various interest rates is shown at figure 5.2.

FIGURE 5.2

Fixed Deposite of Rs.100000/-
Statement showing Amount returned on Daily compounded basis

	PERIOD IN MONTHS						
Rate	12	15	18	21	24	27	30
6.00	106183.13	107787.75	109416.62	111070.10	112748.57	114452.41	116181.99
6.25	106448.88	108125.05	109827.63	111557.02	113313.63	115097.91	116910.28
6.50	106715.28	108463.41	110240.19	112046.06	113881.52	115747.05	117643.13
6.75	106982.36	108802.84	110654.29	112537.25	114452.25	116399.84	118380.57
7.00	107250.10	109143.31	111069.95	113030.59	115025.84	117056.30	119122.62
7.25	107518.51	109484.85	111487.16	113526.08	115602.29	117716.48	119869.33
7.50	107787.59	109827.45	111905.93	114023.74	116181.63	118380.36	120620.70
7.75	108057.34	110171.13	112326.28	114523.59	116763.88	119047.99	121376.78
8.00	108327.76	110515.88	112748.20	115025.62	117349.03	119719.38	122137.60
8.25	108598.85	110861.70	113171.71	115529.84	117937.11	120394.54	122903.18
8.50	108870.63	111208.61	113596.80	116036.27	118528.14	121073.52	123673.55
8.75	109143.08	111556.60	114023.48	116544.93	119122.12	121756.31	124448.75
9.00	109416.21	111905.67	114451.77	117055.80	119719.08	122442.95	125228.80
9.25	109690.03	112255.84	114881.66	117568.91	120319.02	123133.46	126013.73
9.50	109964.52	112607.09	115313.17	118084.27	120921.97	123827.86	126803.59
9.75	110239.71	112959.45	115746.29	118601.89	121527.93	124526.17	127598.38
10.00	110515.58	113312.91	116181.04	119121.77	122136.93	125228.41	128398.15
Rate	33	36	39	42	45	48	51
6.00	117937.71	119719.96	121529.15	123365.68	125229.95	127122.41	129043.45
6.25	118751.20	120621.09	122520.43	124449.68	126409.31	128399.80	130421.62
6.50	119570.27	121528.99	123519.79	125543.20	127599.76	129690.01	131814.50
6.75	120395.00	122443.72	124527.29	126646.32	128801.41	130993.17	133222.23
7.00	121225.41	123365.32	125543.01	127759.13	130014.38	132309.44	134645.00
7.25	122061.55	124293.86	126567.00	128881.71	131238.75	133638.91	136082.94
7.50	122903.44	125229.38	127599.34	130014.14	132474.64	134981.72	137536.23
7.75	123751.13	126171.93	128640.09	131156.52	133722.17	136338.03	139005.05
8.00	124604.67	127121.57	129689.31	132308.92	134981.44	137707.95	140489.53
8.25	125464.09	128078.35	130747.09	133471.44	136252.56	139091.62	141989.84
8.50	126329.42	129042.33	131813.50	134644.17	137535.64	140489.20	143506.19
8.75	127200.73	130013.55	132888.59	135827.20	138830.80	141900.80	145038.70
9.00	128078.03	130992.09	133972.44	137020.61	140138.12	143326.58	146587.58
9.25	128961.38	131977.97	135065.12	138224.50	141457.77	144766.67	148152.98
9.50	129850.81	132971.28	136166.72	139438.95	142789.83	146221.23	149735.09
9.75	130746.38	133972.05	137277.28	140664.08	144134.42	147690.39	151334.08
10.00	131648.11	134980.33	138396.91	141899.95	145491.67	149174.30	152950.14

When bank is applying method of compounding quarterly, it is useful to know the difference of maturity amount with compounding daily.

We have written a C++ program ‘fddiff.cpp’ for the computation of difference of maturity amount on quarterly compounded basis and daily compounded basis. The output for various interest rates is shown at figure 5.3.

FIGURE 5.3

Fixed Deposite of Rs.100000/-
Statement showing difference of maturity amount returned on
Daily and Quarterly compounded basis

	PERIOD IN MONTHS						
Rate	12	15	18	21	24	27	30
6.00	46.78	59.35	72.29	85.61	99.31	113.41	127.91
6.25	50.86	64.57	78.70	93.26	108.26	123.70	139.59
6.50	55.12	70.02	85.41	101.27	117.62	134.48	151.87
6.75	59.57	75.73	92.41	109.63	127.42	145.77	164.72
7.00	64.20	81.66	99.71	118.38	137.66	157.59	178.18
7.25	69.01	87.83	107.31	127.48	148.34	169.92	192.24
7.50	74.00	94.23	115.22	136.95	159.46	182.77	206.91
7.75	79.18	100.90	123.44	146.81	171.05	196.18	222.22
8.00	84.54	107.80	131.96	157.05	183.09	210.12	238.16
8.25	90.09	114.95	140.80	167.67	195.59	224.60	254.74
8.50	95.84	122.35	149.96	178.69	208.58	239.66	271.98
8.75	101.76	130.00	159.43	190.10	222.03	255.28	289.88
9.00	107.88	137.90	169.23	201.91	235.96	271.47	308.45
9.25	114.20	146.06	179.35	214.11	250.39	288.23	327.72
9.50	120.70	154.48	189.80	226.73	265.30	305.60	347.67
9.75	127.40	163.16	200.58	239.76	280.72	323.55	368.32
10.00	134.29	172.09	211.70	253.19	296.64	342.12	389.70
Rate	33	36	39	42	45	48	51
6.00	142.82	158.14	173.91	190.11	206.75	223.85	241.42
6.25	155.97	172.81	190.15	207.98	226.34	245.22	264.63
6.50	169.77	188.23	207.24	226.83	246.99	267.76	289.14
6.75	184.26	204.42	225.20	246.63	268.73	291.51	314.97
7.00	199.44	221.39	244.05	267.45	291.59	316.50	342.19
7.25	215.31	239.16	263.81	289.27	315.58	342.75	370.80
7.50	231.89	257.73	284.48	312.13	340.72	370.28	400.83
7.75	249.20	277.14	306.09	336.05	367.06	399.16	432.36
8.00	267.24	297.39	328.65	361.05	394.61	429.39	465.39
8.25	286.02	318.49	352.19	387.14	423.39	460.98	499.94
8.50	305.56	340.46	376.72	414.36	453.45	494.02	536.09
8.75	325.88	363.33	402.27	442.73	484.80	528.47	573.86
9.00	346.98	387.09	428.83	472.27	517.44	564.44	613.28
9.25	368.87	411.77	456.45	503.00	551.45	601.91	654.41
9.50	391.57	437.38	485.14	534.94	586.84	640.92	697.25
9.75	415.09	463.94	514.91	568.12	623.62	681.53	741.88
10.00	439.44	491.44	545.80	602.56	661.86	723.73	788.31

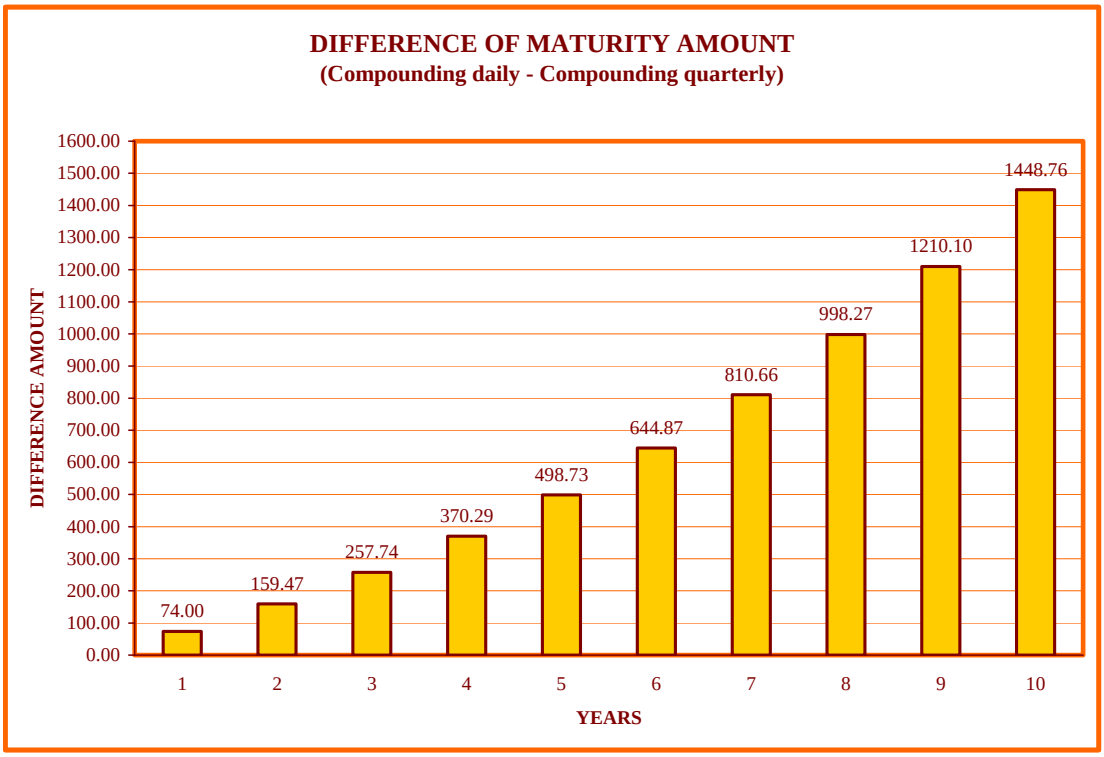
Using worksheet of Excel 'FD-compounding quarterly and daily.xls', following table shows the summary of the difference, on P=1,00,000.00, R=7.5% for the years 1 to 10.

TABLE 5.1 Summary of the difference of maturity amount

No. of Years	Interest Amount (Rs.) Compounding Quarterly	Compounding Monthly	Difference
1	7713.59	7787.58	74.00
2	16022.17	16181.63	159.47
3	24971.64	25229.38	257.74
4	34611.43	34981.72	370.29
5	44994.80	45493.54	498.73
6	56179.10	56823.97	644.87
7	68226.11	69036.77	810.66
8	81202.38	82200.65	998.27
9	95179.58	96389.68	1210.10
10	110234.93	111683.69	1448.76

We can show the above difference graphically as under:

CHART 5.1



From chart 5.1 it is clear that while calculating maturity amount using daily compounding and quarterly compounding methods, the difference of maturity amount increases rapidly with years for a certain rate of interest.

5.3.2 RECURRING TERM DEPOSIT

Recurring term deposit is a very good investment option to the investors who do not want to invest the amount at a time for long term but those who want to invest certain amount every month for certain fixed term.

We have written C++ programs ‘recurqfl.cpp’ and ‘recurdfl.cpp’ to generate a text file output when the principal amount deposited every month is Rs.100 for various rates of interests and various terms using compounding quarterly and compounding daily respectively. We have also written a program called ‘recudiff.cpp’ to generate the difference table of maturity amounts using compounded daily and compounded quarterly. Here generated tabulated text file is showing output of maturity amount for 3, 6, 9, …60 months at different rate of interests.

Calculation of maturity amount for the recurring of Rs.100, duration of n months and quarterly compounding basis or daily compounding basis is given as an example.

for the 1st month, interest will be calculated on Rs.100 for n months. i.e.

$$\text{Interest}_1 = 100 \times \left(1 + \frac{R}{400}\right)^{4 \times \frac{n}{12}} \quad \text{OR} \quad \text{Interest}_1 = 100 \times \left(1 + \frac{R}{36500}\right)^{365 \times \frac{n}{12}}$$

for the 2nd month, interest will be calculated on Rs.100 for (n-1) months. i.e.

$$\text{Interest}_2 = 100 \times \left(1 + \frac{R}{400}\right)^{4 \times \frac{(n-1)}{12}} \quad \text{OR} \quad \text{Interest}_2 = 100 \times \left(1 + \frac{R}{36500}\right)^{365 \times \frac{(n-1)}{12}}$$

Similarly for the nth month, interest will be calculated on Rs.100 for 1 month. i.e.

$$\text{Interest}_n = 100 \times \left(1 + \frac{R}{400}\right)^{4 \times \frac{1}{12}} \quad \text{OR} \quad \text{Interest}_n = 100 \times \left(1 + \frac{R}{36500}\right)^{365 \times \frac{1}{12}}$$

So, Maturity amount after n months=100n + Interest_1 + Interest_2 + …. + Interest_n

Using above mentioned method we have applied the formula of compounding quarterly and generated the output text file using C++ program named ‘recurqfl.cpp’. The generated output is shown at Table 5.2.

TABLE 5.2

Table showing Recurring Deposit Maturity Amount on Compounding Quarterly

(R.D. per month is Rs.100.00)

Rate of Interest	Months									
	3	**6**	**9**	**12**	**15**	**18**	**21**	**24**	**27**	**30**
6.00	303.00	610.54	922.69	1239.52	1561.11	1887.52	2218.83	2555.11	2896.43	3242.87
6.25	303.12	610.98	923.64	1241.19	1563.71	1891.26	2223.93	2561.80	2904.95	3253.46
6.50	303.24	611.42	924.60	1242.87	1566.31	1895.00	2229.04	2568.51	2913.49	3264.08
6.75	303.37	611.86	925.55	1244.54	1568.91	1898.75	2234.16	2575.23	2922.06	3274.74
7.00	303.49	612.30	926.51	1246.21	1571.52	1902.51	2239.30	2581.98	2930.66	3285.44
7.25	303.62	612.74	927.46	1247.89	1574.13	1906.28	2244.44	2588.74	2939.28	3296.17
7.50	303.74	613.18	928.42	1249.57	1576.74	1910.05	2249.60	2595.53	2947.93	3306.95
7.75	303.87	613.62	929.38	1251.25	1579.36	1913.83	2254.77	2602.33	2956.61	3317.76
8.00	303.99	614.06	930.34	1252.93	1581.98	1917.61	2259.96	2609.15	2965.32	3328.62
8.25	304.12	614.50	931.29	1254.62	1584.61	1921.41	2265.15	2615.99	2974.06	3339.51
8.50	304.24	614.95	932.25	1256.30	1587.24	1925.21	2270.36	2622.84	2982.82	3350.45
8.75	304.36	615.39	933.21	1257.99	1589.88	1929.02	2275.58	2629.72	2991.61	3361.42
9.00	304.49	615.83	934.17	1259.68	1592.51	1932.83	2280.81	2636.62	3000.43	3372.43
9.25	304.61	616.27	935.14	1261.37	1595.16	1936.66	2286.06	2643.53	3009.28	3383.48
9.50	304.74	616.71	936.10	1263.07	1597.80	1940.49	2291.31	2650.47	3018.16	3394.57
9.75	304.86	617.16	937.06	1264.76	1600.45	1944.33	2296.58	2657.42	3027.06	3405.71
10.00	304.99	617.60	938.02	1266.46	1603.11	1948.17	2301.86	2664.40	3035.99	3416.88
Rate of Interest	**Months**									
	33	**36**	**39**	**42**	**45**	**48**	**51**	**54**	**57**	**60**
6.00	3594.51	3951.42	4313.69	4681.39	5054.61	5433.42	5817.92	6208.18	6604.30	7006.36
6.25	3607.41	3966.90	4332.00	4702.81	5079.41	5461.89	5850.35	6244.88	6645.58	7052.54
6.50	3620.36	3982.44	4350.40	4724.33	5104.35	5490.54	5883.00	6281.85	6687.17	7099.08
6.75	3633.37	3998.05	4368.89	4745.98	5129.44	5519.36	5915.87	6319.07	6729.07	7146.00
7.00	3646.42	4013.73	4387.46	4767.74	5154.67	5548.37	5948.96	6356.56	6771.29	7193.28
7.25	3659.53	4029.48	4406.13	4789.61	5180.04	5577.55	5982.26	6394.30	6813.82	7240.94
7.50	3672.70	4045.30	4424.89	4811.60	5205.56	5606.91	6015.78	6432.32	6856.67	7288.97
7.75	3685.91	4061.19	4443.75	4833.71	5231.23	5636.45	6049.53	6470.60	6899.84	7337.39
8.00	3699.18	4077.16	4462.69	4855.94	5257.05	5666.18	6083.49	6509.16	6943.33	7386.19
8.25	3712.51	4093.19	4481.73	4878.28	5283.01	5696.09	6117.68	6547.98	6987.15	7435.37
8.50	3725.88	4109.30	4500.86	4900.74	5309.13	5726.19	6152.11	6587.08	7031.29	7484.95
8.75	3739.31	4125.48	4520.08	4923.33	5335.39	5756.46	6186.75	6626.45	7075.77	7534.92
9.00	3752.80	4141.73	4539.40	4946.03	5361.80	5786.93	6221.63	6666.10	7120.58	7585.28
9.25	3766.34	4158.05	4558.82	4968.85	5388.37	5817.59	6256.73	6706.03	7165.73	7636.05
9.50	3779.93	4174.44	4578.32	4991.80	5415.09	5848.44	6292.07	6746.25	7211.21	7687.21
9.75	3793.58	4190.91	4597.93	5014.86	5441.96	5879.47	6327.65	6786.75	7257.04	7738.79
10.00	3807.29	4207.46	4617.63	5038.05	5468.99	5910.70	6363.46	6827.53	7303.20	7790.77

Using above mentioned method we have applied the formula of compounding daily and generated the output text file using C++ program named ‘recurdfl.cpp’. The generated output is shown at Table 5.3.

TABLE 5.3

Table showing Recurring Deposit Maturity Amount on Compounding Daily (R.D. per month is Rs.100.00)

Rate of interest	Months									
	3	**6**	**9**	**12**	**15**	**18**	**21**	**24**	**27**	**30**
6.00	303.02	610.61	922.86	1239.82	1561.58	1888.19	2219.74	2556.30	2897.95	3244.76
6.25	303.14	611.06	923.83	1241.52	1564.21	1891.99	2224.92	2563.10	2906.60	3255.52
6.50	303.27	611.51	924.80	1243.22	1566.85	1895.79	2230.12	2569.92	2915.29	3266.31
6.75	303.40	611.96	925.77	1244.92	1569.50	1899.60	2235.33	2576.76	2924.00	3277.16
7.00	303.52	612.41	926.74	1246.62	1572.15	1903.43	2240.55	2583.63	2932.76	3288.05
7.25	303.65	612.85	927.71	1248.33	1574.81	1907.26	2245.79	2590.51	2941.54	3298.99
7.50	303.78	613.30	928.69	1250.04	1577.47	1911.10	2251.05	2597.43	2950.36	3309.97
7.75	303.90	613.75	929.66	1251.75	1580.14	1914.96	2256.32	2604.36	2959.21	3321.01
8.00	304.03	614.20	930.64	1253.47	1582.82	1918.82	2261.61	2611.32	2968.10	3332.09
8.25	304.16	614.65	931.62	1255.19	1585.50	1922.70	2266.92	2618.31	2977.02	3343.21
8.50	304.29	615.10	932.60	1256.91	1588.19	1926.58	2272.24	2625.32	2985.98	3354.39
8.75	304.41	615.56	933.58	1258.64	1590.88	1930.47	2277.57	2632.35	2994.97	3365.61
9.00	304.54	616.01	934.56	1260.36	1593.58	1934.38	2282.93	2639.41	3004.00	3376.89
9.25	304.67	616.46	935.55	1262.10	1596.28	1938.29	2288.30	2646.49	3013.06	3388.21
9.50	304.79	616.91	936.53	1263.83	1598.99	1942.21	2293.68	2653.59	3022.16	3399.58
9.75	304.92	617.36	937.52	1265.57	1601.71	1946.15	2299.08	2660.72	3031.29	3411.00
10.00	305.05	617.82	938.50	1267.31	1604.43	1950.09	2304.50	2667.88	3040.46	3422.46

Rate of interest	Months									
	33	**36**	**39**	**42**	**45**	**48**	**51**	**54**	**57**	**60**
6.00	3596.81	3954.19	4316.96	4685.21	5059.03	5438.50	5823.70	6214.73	6611.66	7014.59
6.25	3609.92	3969.91	4335.56	4706.98	5084.24	5467.44	5856.68	6252.04	6653.63	7061.55
6.50	3623.09	3985.71	4354.27	4728.87	5109.61	5496.58	5889.89	6289.64	6695.95	7108.91
6.75	3636.32	4001.60	4373.09	4750.90	5135.14	5525.92	5923.35	6327.54	6738.61	7156.68
7.00	3649.61	4017.56	4392.01	4773.06	5160.84	5555.46	5957.05	6365.73	6781.62	7204.86
7.25	3662.97	4033.61	4411.03	4795.35	5186.70	5585.21	5991.00	6404.22	6824.99	7253.46
7.50	3676.39	4049.74	4430.16	4817.78	5212.73	5615.16	6025.20	6443.01	6868.72	7302.49
7.75	3689.87	4065.96	4449.40	4840.34	5238.93	5645.32	6059.66	6482.10	6912.80	7351.93
8.00	3703.42	4082.26	4468.75	4863.04	5265.30	5675.69	6094.36	6521.49	6957.25	7401.81
8.25	3717.03	4098.64	4488.20	4885.88	5291.84	5706.27	6129.32	6561.20	7002.07	7452.13
8.50	3730.71	4115.11	4507.77	4908.86	5318.56	5737.06	6164.54	6601.21	7047.26	7502.88
8.75	3744.45	4131.66	4527.44	4931.97	5345.44	5768.06	6200.02	6641.54	7092.82	7554.08
9.00	3758.26	4148.30	4547.23	4955.22	5372.51	5799.28	6235.77	6682.18	7138.76	7605.72
9.25	3772.13	4165.03	4567.12	4978.62	5399.74	5830.72	6271.77	6723.14	7185.07	7657.81
9.50	3786.06	4181.84	4587.13	5002.16	5427.16	5862.37	6308.04	6764.43	7231.78	7710.36
9.75	3800.07	4198.74	4607.25	5025.84	5454.75	5894.25	6344.59	6806.04	7278.87	7763.37
10.00	3814.14	4215.73	4627.49	5049.66	5482.53	5926.35	6381.40	6847.97	7326.35	7816.84

Using above mentioned method we have applied the formula of compounding daily and compounding quarterly to find the difference of maturity amount and generated the output text file using C++ program named ‘recudiff.cpp’. The generated output is shown at Table 5.4.

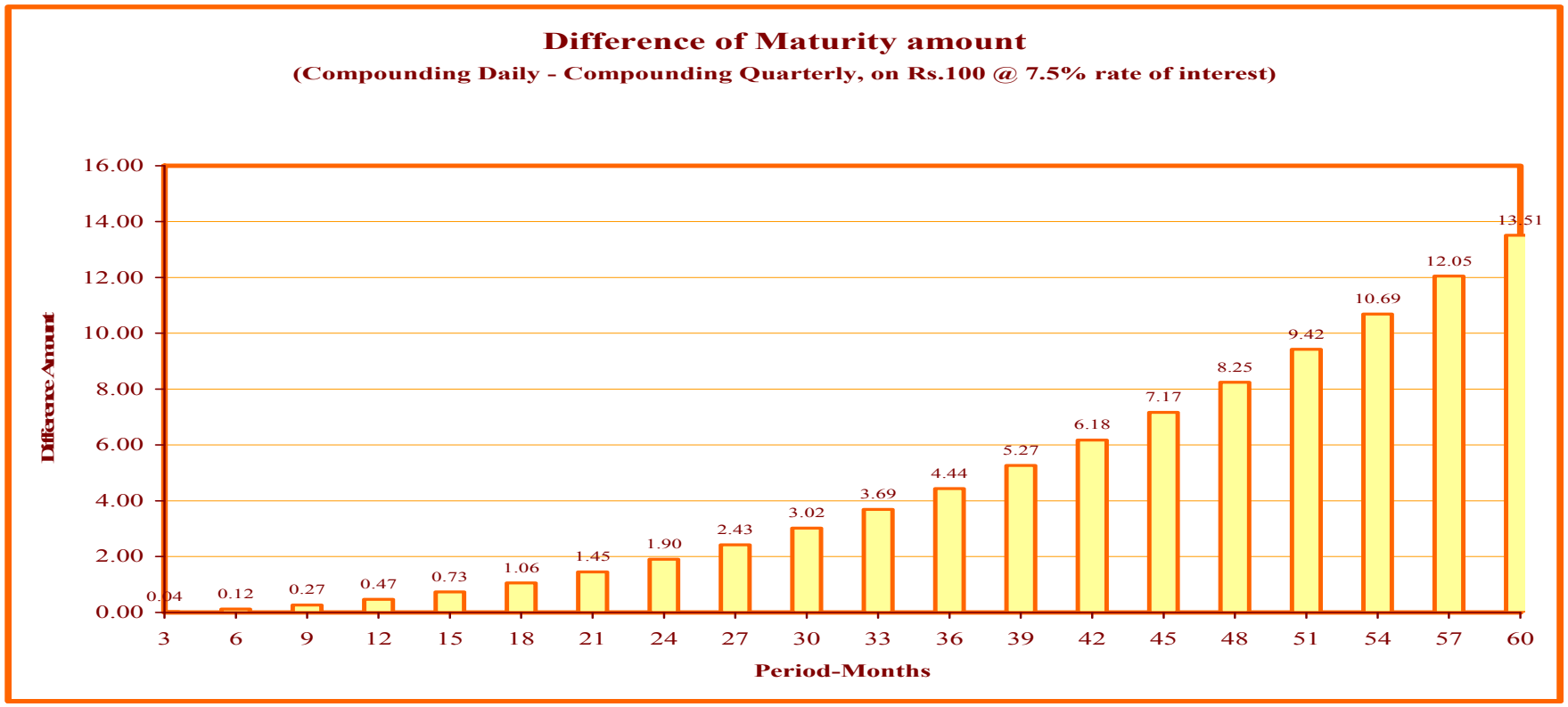
TABLE 5.4

Difference of Recurring Deposit Maturity Value, Deposit Amount Rs.100 P.M., Compounded Daily-Quarterly

Rate of interest	Deposit period in months									
	3	**6**	**9**	**12**	**15**	**18**	**21**	**24**	**27**	**30**
6.00	0.02	0.08	0.17	0.30	0.46	0.67	0.91	1.20	1.52	1.89
6.25	0.02	0.09	0.19	0.32	0.50	0.73	0.99	1.30	1.66	2.06
6.50	0.03	0.09	0.20	0.35	0.55	0.79	1.08	1.41	1.80	2.24
6.75	0.03	0.10	0.22	0.38	0.59	0.85	1.16	1.53	1.95	2.42
7.00	0.03	0.11	0.23	0.41	0.64	0.92	1.25	1.65	2.10	2.61
7.25	0.03	0.12	0.25	0.44	0.68	0.99	1.35	1.77	2.26	2.81
7.50	0.04	0.12	0.27	0.47	0.73	1.06	1.45	1.90	2.43	3.02
7.75	0.04	0.13	0.29	0.50	0.78	1.13	1.55	2.04	2.60	3.24
8.00	0.04	0.14	0.31	0.54	0.84	1.21	1.65	2.18	2.78	3.47
8.25	0.04	0.15	0.33	0.57	0.89	1.29	1.76	2.32	2.97	3.70
8.50	0.05	0.16	0.35	0.61	0.95	1.37	1.88	2.47	3.16	3.94
8.75	0.05	0.17	0.37	0.64	1.01	1.45	1.99	2.63	3.36	4.20
9.00	0.05	0.18	0.39	0.68	1.07	1.54	2.12	2.79	3.57	4.46
9.25	0.05	0.19	0.41	0.72	1.13	1.63	2.24	2.95	3.78	4.72
9.50	0.06	0.20	0.43	0.76	1.19	1.73	2.37	3.13	4.00	5.00
9.75	0.06	0.21	0.46	0.80	1.26	1.82	2.50	3.30	4.23	5.29
10.00	0.06	0.22	0.48	0.85	1.33	1.92	2.64	3.49	4.47	5.59
Rate of interest	**Deposit period in months**									
	33	**36**	**39**	**42**	**45**	**48**	**51**	**54**	**57**	**60**
6.00	2.30	2.76	3.27	3.82	4.43	5.08	5.79	6.55	7.36	8.24
6.25	2.51	3.01	3.57	4.17	4.83	5.55	6.32	7.16	8.05	9.01
6.50	2.73	3.27	3.88	4.54	5.26	6.04	6.89	7.80	8.78	9.82
6.75	2.95	3.55	4.20	4.92	5.70	6.56	7.48	8.47	9.54	10.68
7.00	3.19	3.83	4.54	5.32	6.17	7.10	8.10	9.18	10.34	11.58
7.25	3.44	4.13	4.90	5.74	6.66	7.66	8.74	9.91	11.17	12.52
7.50	3.69	4.44	5.27	6.18	7.17	8.25	9.42	10.69	12.05	13.51
7.75	3.96	4.76	5.65	6.63	7.70	8.87	10.13	11.50	12.97	14.55
8.00	4.24	5.10	6.06	7.11	8.26	9.51	10.87	12.34	13.92	15.63
8.25	4.53	5.45	6.47	7.60	8.83	10.18	11.64	13.22	14.93	16.76
8.50	4.83	5.81	6.91	8.11	9.43	10.87	12.44	14.13	15.96	17.93
8.75	5.14	6.19	7.36	8.64	10.06	11.60	13.27	15.09	17.05	19.16
9.00	5.46	6.58	7.82	9.20	10.70	12.35	14.14	16.08	18.18	20.44
9.25	5.79	6.98	8.31	9.77	11.37	13.13	15.04	17.11	19.35	21.76
9.50	6.13	7.40	8.81	10.36	12.07	13.94	15.97	18.18	20.57	23.14
9.75	6.49	7.83	9.32	10.97	12.79	14.77	16.94	19.29	21.83	24.58
10.00	6.85	8.28	9.86	11.61	13.53	15.64	17.94	20.44	23.15	26.07

From above table 5.4, it is clear that the maturity amount when calculated by compounded daily is more than the maturity amount when calculated by compounded quarterly. For a certain rate of interest the difference of maturity amount by both the methods increases rapidly with the tenure.

CHART 5.2



From chart 5.2 it is clear that while calculating maturity amount using daily compounding and quarterly compounding methods, the difference of maturity amount increases rapidly with tenure for a certain rate of interest.

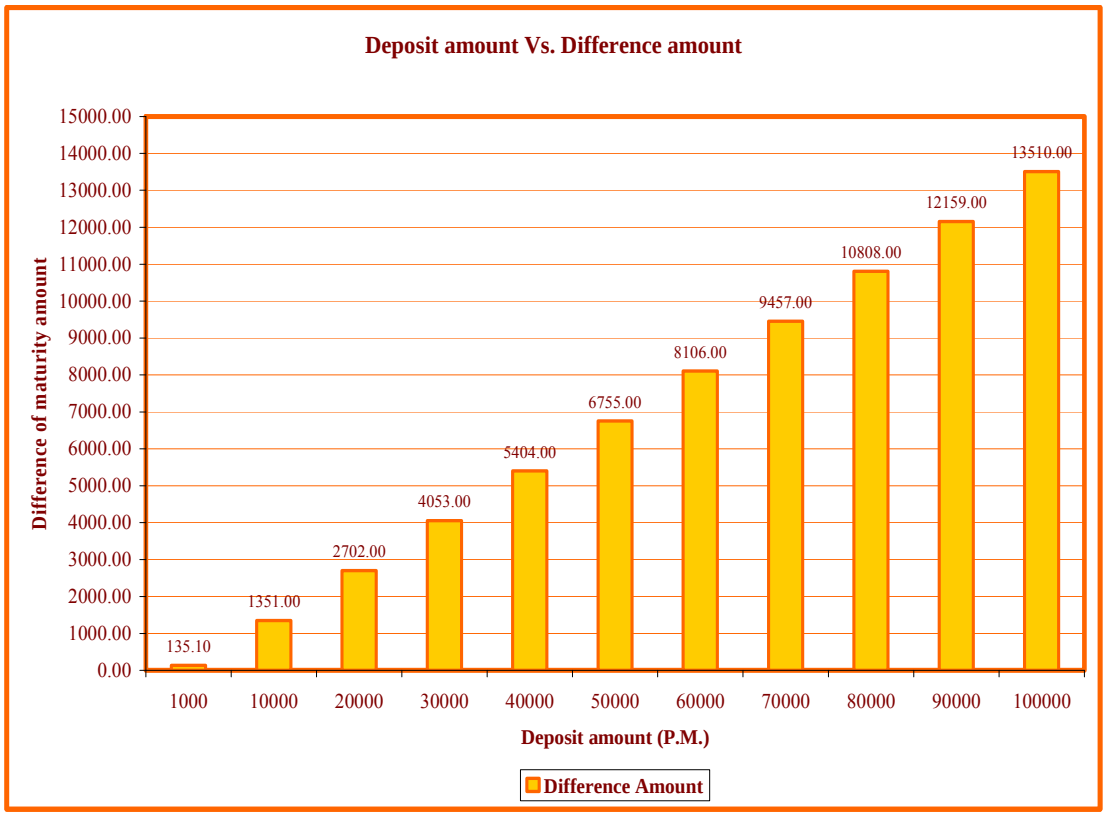
We have calculated the difference of maturity amounts for the recurring of Rs.100. We can obtain the same for any amount of recurring deposit. Let us calculate the differences for various amounts of deposit amounts per month for the duration of 5 years at 7.5% rate of interest. Summary is shown in Table 5.5.

TABLE 5.5

Deposit amount P.M.	Difference of Maturity Amount
100	13.51
1000	135.10
10000	1351.00
20000	2702.00
30000	4053.00
40000	5404.00
50000	6755.00
60000	8106.00
70000	9457.00
80000	10808.00
90000	12159.00
100000	13510.00

The graph of Deposit amount P.M. Vs. Difference amount is shown at chart 5.3.

CHART 5.3



From chart 5.3, it is clear that for the certain rate of interest and tenure, the difference of maturity amount increases rapidly with the amount deposited per month.

5.4 DIFFERENCE OF POLICY

While giving loans to the customer, bank is calculating interest on daily compounding basis, which is comparatively higher than quarterly compounding basis. The same bank is calculating interest on quarterly compounding basis while taking money as fixed deposit or recurring deposit.

Chapter

6

Weibull Distribution for Amount Withdrawn Per Day

6.1 INTRODUCTION

In the banking services ATM is playing very important role because it helps to solve problems for which people do not have to come to bank and wait to withdraw money. ATM machine contains a paper role in it, on which details of each visit of a customer is printed. Details are like date, time, amount withdrawn etc. Here in this chapter we have focused our study on the amount withdrawn per day from the ATM. We have collected data of hit and amount withdrawn through ATM for 1 year varying from 01-01-2005 to 31-12-2005.

In this chapter our intension is to fit this data for a distribution. For which we obtained frequency distribution of amount withdrawn per day with class interval 25000 and then we draw the graph for number of amount Vs. frequency and found that frequency distribution is right skewed. Hence in such situation we found that weibull distribution will fit well.

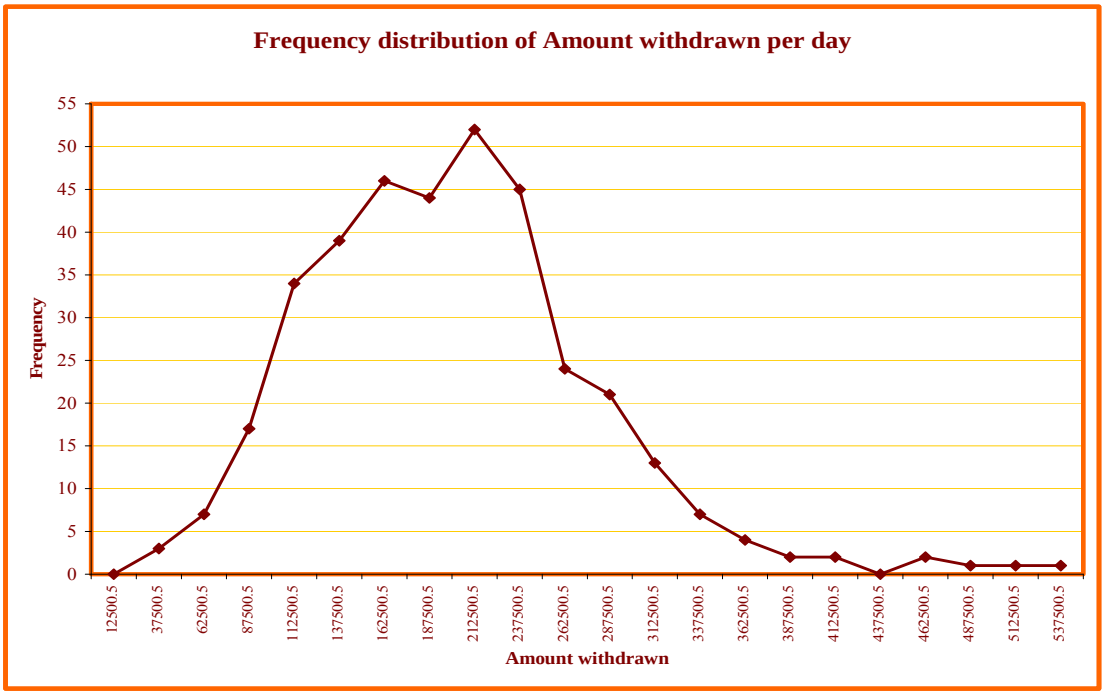
6.2 FREQUENCY DISTRIBUTION OF AMOUNT WITHDRAWN PER DAY

TABLE 6.1

Lower Limit	Upper Limit	Mid value	Frequency
1	25000	12500.5	0
25001	50000	37500.5	3
50001	75000	62500.5	7
75001	100000	87500.5	17
100001	125000	112500.5	34

125001	150000	137500.5	39
150001	175000	162500.5	46
175001	200000	187500.5	44
200001	225000	212500.5	52
225001	250000	237500.5	45
250001	275000	262500.5	24
275001	300000	287500.5	21
300001	325000	312500.5	13
325001	350000	337500.5	7
350001	375000	362500.5	4
375001	400000	387500.5	2
400001	425000	412500.5	2
425001	450000	437500.5	0
450001	475000	462500.5	2
475001	500000	487500.5	1
500001	525000	512500.5	1
525001	550000	537500.5	1

CHART 6.1



From the above charts we observed that frequency distribution is right skewed, in such a situation we anticipate that weibull distribution will fit well.

6.3 DEFINITION

The life time random variable x is said to have Weibull distribution if its probability density function is given by

$$f(x,\beta,\theta)=\frac{\beta}{\theta}x^{\beta-1}\exp\left(-\frac{x^{\beta}}{\theta}\right) \text{ where } x>0,\ \beta>0,\ \theta>0 \tag{6.1}$$

β is the shape parameter and θ is the scale parameter, which is known as two-parameter Weibull distribution.

6.3.1 MAXIMUM LIKELIHOOD ESTIMATION

Cohen (1965) suggested the method of Maximum Likelihood Estimation to estimate β and θ by means of iterative procedure.

Let $x_1, x_2, x_3, \ldots, x_n$ be random sample of size n from w(0, β, θ) distribution. The Likelihood function of this sample is

$$L(x_1, x_2, x_3, \ldots, x_n; \beta\ \theta)= \prod_{i=1}^{n}\frac{\beta}{\theta}x_i^{\beta-1}\exp\left\{-\frac{x_i^{\beta}}{\theta}\right\}$$

Taking logarithm we get, $\text{Ln } L= n \ln \beta - n \ln \theta + (\beta-1)\sum_{i=1}^{n}\ln x_i - \frac{\sum_{i=1}^{n}x_i^{\beta}}{\theta}$ (6.2)

Differentiating log likelihood with respect to the parameters β and θ and equating to zero we obtain

$$\frac{\partial \ln L}{\partial \beta}=\frac{n}{\beta}+\sum_{i=1}^{n}\ln x_i-\frac{\sum_{i=1}^{n}x_i^{\beta}\ln x_i}{\theta}=0 \tag{6.3}$$

$$\frac{\partial \ln L}{\partial \theta}=-\frac{n}{\beta}+\frac{\sum_{i=1}^{n}x^{\beta}}{\theta^2}=0 \tag{6.4}$$

$$\frac{\partial^2 \ln L}{\partial \beta^2}=\frac{-n}{\beta^2}-\frac{n\left(\sum_{i=1}^{n}x_i^{\beta}(\ln x_i)^2\right)}{\left(\sum_{i=1}^{n}x_i^{\beta}\right)} \tag{6.5}$$

$$\frac{\partial^2 \ln L}{\partial \theta^2} = \frac{n}{\theta^2} - \frac{2}{\theta^3}\sum_{i=1}^{n}\left(x_i^{\beta}\right) \tag{6.6}$$

$$\frac{\partial^2 \ln L}{\partial \beta \partial \theta} = \frac{\partial^2 \ln L}{\partial \theta \partial \beta} = \frac{1}{\theta^2}\sum_{i=1}^{n}\left(x_i^{\beta} \ln x_i\right) \tag{6.7}$$

$$I(\theta) = E\begin{pmatrix} -\frac{\partial^2 \ln L}{\partial \beta^2} & -\frac{\partial^2 \ln L}{\partial \beta \partial \theta} \\ -\frac{\partial^2 \ln L}{\partial \theta \partial \beta} & -\frac{\partial^2 \ln L}{\partial \theta^2} \end{pmatrix}_{(\hat{\beta},\hat{\theta})} \tag{6.8}$$

$$\sum = E\left(I(\theta)\right)^{-1}$$

$$K = E\left(-\frac{\partial^2 \ln L}{\partial \hat{\beta}^2}\right)E\left(-\frac{\partial^2 \ln L}{\partial \hat{\theta}^2}\right) - E\left(-\frac{\partial^2 \ln L}{\partial \hat{\beta} \partial \hat{\theta}}\right)E\left(-\frac{\partial^2 \ln L}{\partial \hat{\theta} \partial \hat{\beta}}\right)$$

$$\sum = \frac{1}{K}\begin{pmatrix} E\left(-\frac{\partial^2 \ln L}{\partial \theta^2}\right) & E\left(-\frac{\partial^2 \ln L}{\partial \beta \partial \theta}\right) \\ E\left(-\frac{\partial^2 \ln L}{\partial \theta \partial \beta}\right) & E\left(-\frac{\partial^2 \ln L}{\partial \beta^2}\right) \end{pmatrix}_{(\hat{\beta},\hat{\theta})}$$

6.4 INITIAL SOLUTION USING GRAPHICAL METHOD

We have, $\overline{F}(x) = e^{\frac{-x^{\beta}}{\theta}}$

$$\ln \overline{F}(x) = \frac{-x^{\beta}}{\theta}$$

$$-\ln \overline{F}(x) = \frac{x^{\beta}}{\theta}$$

$$\ln\left(-\ln \overline{F}(x)\right) = -\ln\theta + \beta \ln x$$

$y = \mu + \beta t$

Where, t=ln x and μ = -ln θ

For given data $x_1, x_2, x_3, \ldots, x_n$ we obtain

$t_i = \ln x_i$, i=1,2,3,…. (6.9)

The survival function $\overline{F}(x)$ is calculated using Kaplan-meier estimator as

$$\hat{\bar{F}}\left(x_{(i)}\right) = \frac{n-i+1}{n+1}$$

Define $s_i = \ln(-\ln \bar{F}(x_{(i)}))$

$$= \ln(\ln(\frac{n+1}{n-i+1})) \tag{6.10}$$

By taking (s_i, t_i), i=1, 2, 3, …, n, we fit a straight line.

Then estimator of slope (which is same as shape parameter)

$$\hat{\beta} = \frac{\sum(s_i - \bar{s})(t_i - \bar{t})}{\sum(t_i - \bar{t})^2} \tag{6.11}$$

And

$$\hat{\theta} = \begin{cases} \dfrac{\sum x_i^{\beta}}{n}, (uncensored) \\ \dfrac{\sum x_i^{\hat{\beta}} + (n-r)x_{(r)}^{\hat{\beta}}}{r}, (censored) \end{cases} \tag{6.12}$$

To calculate initial solution, we have developed a C++ program named 'gramt.cpp'. We are using data files to handle the data. A data file 'snramta.csv' consisting fields like serial number and data observations $x_1, x_2, \ldots, x_n$ (showing amount withdrawn per day) is created in Excel. We are reading the same and creating one data file to store our tabulated output. Following is the algorithm for the program:

1. Open input data file 'snramta.csv' to read and output data file 'graphamt.txt' to write.
2. Read the input data file 'snramta.csv' till end of file and do following till end of file.
   a. Store observation x to array element ax[i] and add to tx for total of x.
   b. Calculate $s_i = \ln\left[\ln\left(\frac{n+1}{n-i+1}\right)\right]$ and store in array element as[i] and add to ts for total.
   c. Calculate $t_i = \ln(x_i)$ and store in array element at[i] and add to tt for total.

3. Calculate sbar, $\bar{s} = \frac{ts}{n}$

4. Calculate tbar, $\bar{t} = \frac{tt}{n}$

5. Repeat following steps n times for i=1,2,3,…,n.

   a. Generate array element $ab_{01}[i] = (as[i] - \bar{s})(at[i] - \bar{t})$ and add to tb_{01} for total.

   b. Generate array element $ab_{02}[i] = (at[i] - \bar{t})^2$ and add to tb_{02} for total.

6. Calculate initial solution of β as $b_0 = \frac{tb_{01}}{tb_{02}}$

7. Generate output file write column headings to output file.

8. Repeat following steps n times for i=1,2,3,….,n.

   a. Generate array element $ay4[i] = ax[i]^{b0}$ and add to ty4 for total.

   b. Write i, ax[i], as[i], at[i], ab_{01}[i], ab_4[i], ay4[i] to output file.

9. Write column wise totals like tx, ts, tt, tb_{01}, tb_{02}, ty4 to output file.

10. Write values of sbar and tbar to output file.

11. calculate $t_0 = \frac{\text{ty4}}{n}$

12. Write value of β (b_0) and θ (t_0) to output file.

13. Close data files and terminate the process.

Following is the created output file 'graphamt.txt':

TABLE 6.2 Initial Solution of $\hat{\beta}$ and $\hat{\theta}$ Using Graphical Method

i Day number	x Amount withdrawn	$s_i = \ln\left[\ln\left(\frac{n+1}{n-i+1}\right)\right]$	$t_i = \ln(x_i)$	$(s_i - \bar{s})^* (t_i - \bar{t})$	$(t_i - \bar{t})^2$	$x_i^{b_0}$
1	17000	-5.901266	9.740969	11.822866	4.919790	21212127949.268600
2	17300	-5.206748	9.758462	10.201290	4.842494	22137524948.532500
3	20000	-4.799909	9.903488	8.692712	4.225249	31540866128.134400
4	33700	-4.510849	10.425253	6.042857	2.352467	112721591783.284000
5	34300	-4.286326	10.442901	5.632921	2.298644	117683500445.380000
6	35800	-4.102621	10.485703	5.203238	2.170688	130644567516.230000

7	36800	-3.947083	10.513253	4.881070	2.090267	139732547901.479000
8	37100	-3.812162	10.521372	4.659688	2.066856	142529517451.036000
9	40800	-3.692986	10.616437	4.191561	1.802551	179756717809.992000
10	49100	-3.586229	10.801614	3.489878	1.339608	282484552995.590000
11	51000	-3.489519	10.839581	3.267138	1.253163	309915865573.432000
12	51100	-3.401105	10.841540	3.162619	1.248781	311401312593.365000
13	51500	-3.319656	10.849337	3.050169	1.231415	317385085389.814000
14	51600	-3.244138	10.851277	2.961182	1.227114	318891542830.317000
15	53700	-3.173733	10.891168	2.779363	1.140326	351505844663.048000
16	53900	-3.107778	10.894886	2.699502	1.132400	354710069486.427000
17	59200	-3.045734	10.988677	2.401370	0.941582	445967479080.518000
18	60400	-2.987152	11.008744	2.296039	0.903040	468357189873.745000
19	61500	-2.931658	11.026792	2.200699	0.869064	489452206015.519000
20	62500	-2.878935	11.042922	2.114322	0.839251	509107332086.739000
21	63500	-2.828712	11.058795	2.032475	0.810420	529220901227.933000
22	67200	-2.780755	11.115429	1.864156	0.711661	607680163307.128000
23	67800	-2.734863	11.124317	1.806207	0.696743	621009723587.178000
24	70800	-2.690860	11.167614	1.677694	0.626337	690236878270.860000
25	72100	-2.648591	11.185809	1.606439	0.597868	721584286929.412000
26	72600	-2.607919	11.192720	1.560914	0.587228	733860341961.202000
27	74800	-2.568725	11.222573	1.471241	0.542367	789334539968.096000
28	76700	-2.530899	11.247657	1.394222	0.506049	839175779551.080000
29	77300	-2.494346	11.255449	1.353232	0.495024	855290500917.034000
30	78600	-2.458980	11.272127	1.296862	0.471834	890828368896.721000
31	79500	-2.424721	11.283512	1.252224	0.456322	915933267726.770000
32	79800	-2.391500	11.287279	1.222926	0.451248	924393222611.488000
33	79900	-2.359252	11.288531	1.199024	0.449567	927223416009.026000
34	83000	-2.327920	11.326596	1.111138	0.399971	1017507348009.160000
35	83200	-2.297449	11.329003	1.087713	0.396933	1023502681031.850000
36	83500	-2.267791	11.332602	1.062920	0.392410	1032534694398.160000
37	83600	-2.238901	11.333799	1.042826	0.390912	1035555780226.940000
38	85100	-2.210738	11.351582	0.996058	0.368991	1081499036167.050000
39	85800	-2.183264	11.359774	0.966161	0.359106	1103343140487.960000
40	86000	-2.156444	11.362103	0.946397	0.356321	1109631720343.000000
41	87800	-2.130245	11.382817	0.898460	0.332020	1167181462135.830000
42	88200	-2.104638	11.387362	0.876734	0.326802	1180204079418.500000
43	88600	-2.079593	11.391887	0.855590	0.321649	1193312081020.270000
44	89500	-2.055086	11.401994	0.826692	0.310288	1223118153784.400000
45	89900	-2.031092	11.406453	0.806815	0.305339	1236504830120.110000
46	90900	-2.007587	11.417515	0.777935	0.293237	1270348604784.470000
47	92600	-1.984552	11.436044	0.739269	0.273512	1329125546983.620000
48	94100	-1.961965	11.452113	0.705105	0.256963	1382295691239.250000
49	95900	-1.939808	11.471061	0.667937	0.238112	1447731477592.090000
50	96500	-1.918064	11.477298	0.648925	0.232064	1469941348219.070000

.
.
.

315	252100	0.678443	12.437581	0.597921	0.229013	15321945389258.900000
316	253000	0.688441	12.441145	0.607193	0.232436	15455811319606.000000
317	253600	0.698539	12.443514	0.615069	0.234726	15545437416127.800000
318	254200	0.708741	12.445877	0.623036	0.237021	15635369599927.000000
319	255400	0.719052	12.450586	0.634131	0.241629	15816153505589.000000
320	255600	0.729475	12.451369	0.640273	0.242399	15846403493579.100000
321	256500	0.740017	12.454884	0.650071	0.245873	15982950940112.300000
322	256600	0.750682	12.455274	0.655874	0.246259	15998165584856.800000
323	263800	0.761475	12.482947	0.698104	0.274490	17116171572899.400000
324	268600	0.772404	12.500979	0.728053	0.293710	17886391826638.500000
325	269300	0.783473	12.503581	0.737577	0.296538	18000390563084.300000
326	270000	0.794690	12.506177	0.747231	0.299372	18114817102327.800000
327	270100	0.806061	12.506548	0.753963	0.299777	18131198702051.900000
328	271300	0.817596	12.510980	0.766434	0.304651	18328460190454.500000
329	271800	0.829301	12.512822	0.775473	0.306687	18411024602258.500000
330	272800	0.841185	12.516494	0.787240	0.310768	18576811077699.800000
331	273200	0.853259	12.517959	0.796058	0.312404	18643371443219.600000
332	273700	0.865533	12.519788	0.805545	0.314451	18726769600407.500000
333	275300	0.878018	12.525617	0.820992	0.321022	18995121848974.800000
334	275700	0.890726	12.527069	0.830314	0.322670	19062562322080.700000
335	276600	0.903670	12.530328	0.842473	0.326383	19214819586196.300000
336	277300	0.916865	12.532855	0.853772	0.329277	19333736568800.400000
337	279100	0.930327	12.539325	0.871211	0.336744	19641514180724.600000
338	279600	0.944073	12.541115	0.881899	0.338825	19727517602094.300000
339	280500	0.958122	12.544329	0.894991	0.342577	19882883090787.100000
340	284100	0.972496	12.557082	0.923088	0.357667	20511555289163.300000
341	286000	0.987218	12.563747	0.942278	0.365685	20848021072175.900000
342	286200	1.002314	12.564446	0.952507	0.366530	20883626632391.300000
343	290200	1.017814	12.578326	0.983943	0.383529	21603285795575.200000
344	299200	1.033750	12.608868	1.042824	0.422291	23275437890389.900000
345	299800	1.050161	12.610871	1.056736	0.424898	23389537717750.300000
346	302100	1.067088	12.618513	1.080289	0.434920	23829975690601.900000
347	302200	1.084579	12.618844	1.092372	0.435357	23849235264837.800000
348	308900	1.102691	12.640773	1.141024	0.464775	25160619301038.800000
349	310300	1.121489	12.645295	1.161492	0.470961	25439885584293.100000
350	313500	1.141047	12.655555	1.192480	0.485149	26085054523362.500000
351	318900	1.161457	12.672633	1.236283	0.509231	27195477273588.700000
352	324900	1.182824	12.691273	1.284221	0.536182	28461459093804.800000
353	334000	1.205277	12.718896	1.349730	0.577399	30446782615275.700000
354	335400	1.228976	12.723079	1.375266	0.583773	30759249248718.100000
355	346000	1.254116	12.754194	1.451263	0.632288	33186484539162.000000
356	349000	1.280947	12.762827	1.488586	0.646092	33893265344453.100000
357	369400	1.309794	12.819635	1.618617	0.740644	38934696737817.800000
358	373100	1.341086	12.829602	1.664604	0.757898	39893521828098.300000
359	380730	1.375416	12.849846	1.733894	0.793555	41914414564391.700000
360	382900	1.413636	12.855529	1.779220	0.803713	42499955925679.300000
361	440900	1.457031	12.996573	2.104165	1.076499	59966973053443.700000
362	446300	1.507702	13.008747	2.182042	1.101908	61775639528712.800000

363	457300	1.569453	13.033095	2.298980	1.153619	65558543733497.000000
364	488700	1.650481	13.099504	2.533536	1.300684	77095834415330.200000
365	523100	1.775399	13.167528	2.835612	1.460471	91021559723153.200000
TOT	63822850	-208.412057	4365.045414	232.581184	95.280569	3131158681713497.280000
		-0.570992	11.959029			

Remark: We did not show the entire value of x because of space. If any one require, we can provide it as we have stored as soft copy.

As per given algorithm, we have calculated values of initial solution. i.e. β and θ.

β=2.4410138 and θ=8578516936201.362400.

6.5 ESTIMATION OF PARAMETERS BY N-R METHOD

We have written a C++ program 'nramt.cpp', to get the final solution using N-R method. Output is stored in a data file named 'nramt.txt'.

To write the algorithm let us rewrite equations (6.3) and (6.5) for the reference, as

$$\frac{\partial \ln L}{\partial \beta} = \frac{n}{\beta} + \sum_{i=1}^{n} \ln x_i - \frac{\sum_{i=1}^{n} x_i^{\beta} \ln x_i}{\theta} = 0 \qquad (6.3)$$

Equation (6.3) can be rewritten as

$$z_1 = \frac{n}{b_0} + y_1 - \frac{y_2}{t_0}$$

Where, $\beta = b_0$, $\theta = t_0$, $y_1 = \sum_{i=1}^{n} \ln x_i$ and $y_2 = \sum_{i=1}^{n} x_i^{\beta} \ln x_i$

$$\frac{\partial^2 \ln L}{\partial \beta^2} = \frac{-n}{\beta^2} - \frac{n\left(\sum_{i=1}^{n} x_i^{\beta} (\ln x_i)^2\right)}{\left(\sum_{i=1}^{n} x_i^{\beta}\right)} \qquad (6.5)$$

Equation (6.5) can be rewritten as

$Z_2 = -\frac{n}{b_0^{\,2}} - \frac{n y_3}{y_4}$ Where, $\beta = b_0$, $y_3 = \sum_{i=1}^{n} x_i^{\beta} (\ln x_i)^2$ and $y_4 = \sum_{i=1}^{n} x_i^{\beta}$

Algorithm for the above program 'nramt.cpp' is as under:

1. Find the initial solution, i.e., calculate b_0 and t_0 using graphical method as shown earlier.

2. Calculate $y_1 = \sum_{i=1}^{n} \ln x_i$

3. Calculate $y_2 = \sum_{i=1}^{n} x_i^{b_0} \ln x_i$

4. Calculate $y_3 = \sum_{i=1}^{n} x_i^{b_0} (\ln x_i)^2$

5. Calculate $y_4 = \sum_{i=1}^{n} x_i^{b_0}$

6. Calculate $z_1 = \frac{n}{b_0} + y_1 - \frac{y_2}{t_0}$

7. Calculate $z_2 = -\frac{n}{b_0^{\,2}} - \frac{n y_3}{y_4}$

8. Calculate $b_1 = b_0 - \frac{z_1}{z_2}$

9. Calculate $t_1 = \frac{y_4}{n}$

10. Calculate the difference between b_0 and b_1, i.e. difference= abs(b_1-b_0)

11. Assign value of b_1 to b_0

12. Repeat above steps 3 to 10 till the difference<= 0.0001

13. Terminate the process.

This is iterative process. To reduce the number of iterations we have taken the initial solution by graphical method, i.e., nearer to the final solution. Using C++ program we could get the final solution (with the difference > 0.0001) after 343 iterations. Portion of tabulated output and solution of last iteration by N-R method using 'nramt.cpp' program is as under:

TABLE 6.3 Shows the portion of last iteration no. 343 using N-R method

i Day number	x Amount withdrawn	s_i	t_i	b_{01}	b_{02}	y_1	y_2	y_3	y_4
1	17000	-5.90	9.74	11.82	4.92	9.74	127127987156.54	1238349734001.87	13050856857.90
2	17300	-5.21	9.76	10.20	4.84	9.76	132796428479.02	1295888871897.54	13608336176.98
3	20000	-4.80	9.90	8.69	4.23	9.90	190632586466.48	1887927447178.51	19249035802.31
4	33700	-4.51	10.43	6.04	2.35	10.43	698763175776.61	7284782975849.02	67026015385.79
5	34300	-4.29	10.44	5.63	2.30	10.44	730114287438.19	7624510954563.03	69914893676.21
6	35800	-4.10	10.49	5.20	2.17	10.49	812112378004.96	8515569338382.78	77449491431.67
7	36800	-3.95	10.51	4.88	2.09	10.51	869691662791.03	9143288590891.08	82723363788.58
8	37100	-3.81	10.52	4.66	2.07	10.52	887425666568.35	9336935780923.66	84345049828.15
9	40800	-3.69	10.62	4.19	1.80	10.62	1123983760178.36	11932703184031.70	105872028630.97
10	49100	-3.59	10.80	3.49	1.34	10.80	1780612355166.84	19233487902868.40	164846874128.33
11	51000	-3.49	10.84	3.27	1.25	10.84	1956681454507.45	21209606944569.00	180512648085.34
12	51100	-3.40	10.84	3.16	1.25	10.84	1966223189295.27	21316886915615.40	181360141629.80
13	51500	-3.32	10.85	3.05	1.23	10.85	2004667134310.86	21749309486670.90	184773237138.82
14	51600	-3.24	10.85	2.96	1.23	10.85	2014347500286.81	21858242602112.80	185632300170.35
15	53700	-3.17	10.89	2.78	1.14	10.89	2224096865989.71	24223013239619.80	204211045932.74
16	53900	-3.11	10.89	2.70	1.13	10.89	2244721019849.31	24455979067363.80	206034378876.19
17	59200	-3.05	10.99	2.40	0.94	10.99	2833242039848.47	31133581131203.30	257832866143.35
18	60400	-2.99	11.01	2.30	0.90	11.01	2977936913690.83	32783346274270.60	270506500090.94
19	61500	-2.93	11.03	2.20	0.87	11.03	3114362526173.54	34341429201991.10	282435943110.71
20	62500	-2.88	11.04	2.11	0.84	11.04	3241557539809.71	35796266538122.00	293541653923.79

21	63500	-2.83	11.06	2.03	0.81	11.06	3371797180538.71	37288014424536.10	304897335032.35
22	67200	-2.78	11.12	1.86	0.71	11.12	3880533966813.36	43133797952813.00	349112403319.22
23	67800	-2.73	11.12	1.81	0.70	11.12	3967066673123.00	44130909112061.50	356612141142.18
24	70800	-2.69	11.17	1.68	0.63	11.17	4416911289173.90	49326361585066.20	395510731169.32
25	72100	-2.65	11.19	1.61	0.60	11.19	4620836153992.47	51687792132646.20	413098062057.83
26	72600	-2.61	11.19	1.56	0.59	11.19	4700732067827.87	52613978774186.00	419981200592.01
27	74800	-2.57	11.22	1.47	0.54	11.22	5062015895923.25	56808843749140.30	451056617940.19
28	76700	-2.53	11.25	1.39	0.51	11.25	5386935235389.91	60590399740764.10	478938435039.93
29	77300	-2.49	11.26	1.35	0.50	11.26	5492050064704.47	61815490697028.40	487945878502.43
30	78600	-2.46	11.27	1.30	0.47	11.27	5723962137404.46	64521228032476.00	507797875978.87
31	79500	-2.42	11.28	1.25	0.46	11.28	5887872903018.84	66435886325832.30	521812069339.76
32	79800	-2.39	11.29	1.22	0.45	11.29	5943123103155.11	67081687309602.30	526532853239.67
33	79900	-2.36	11.29	1.20	0.45	11.29	5961608154018.83	67297799241962.20	528111946934.26
34	83000	-2.33	11.33	1.11	0.40	11.33	6551703749965.82	74208500745755.70	578435375946.76
35	83200	-2.30	11.33	1.09	0.40	11.33	6590916877834.52	74668514622069.10	581773797301.24
36	83500	-2.27	11.33	1.06	0.39	11.33	6649998002478.27	75361780069960.50	586802400260.61
37	83600	-2.24	11.33	1.04	0.39	11.33	6669761505032.24	75593734935836.40	588484198747.02
38	85100	-2.21	11.35	1.00	0.37	11.35	6970416646692.46	79125258331719.00	614048020226.31
39	85800	-2.18	11.36	0.97	0.36	11.36	7113430608530.01	80806966108297.50	626194713888.36
40	86000	-2.16	11.36	0.95	0.36	11.36	7154609690855.01	81291409193269.60	629690644269.39
41	87800	-2.13	11.38	0.90	0.33	11.38	7531611822494.00	85730957430693.20	661665031451.32
42	88200	-2.10	11.39	0.88	0.33	11.39	7616958955532.48	86737070809055.40	668895815700.17
43	88600	-2.08	11.39	0.86	0.32	11.39	7702879252644.21	87750331072928.00	676172363743.05
44	89500	-2.06	11.40	0.83	0.31	11.40	7898301881141.90	90056389902808.60	692712340267.86

45	89900	-2.03	11.41	0.81	0.31	11.41	7986093390955.14	91093000678152.20	700138179379.94
46	90900	-2.01	11.42	0.78	0.29	11.42	8208105051286.60	93716164844268.90	718904669700.39
47	92600	-1.98	11.44	0.74	0.27	11.44	8593873797890.11	98279922497996.30	751472579311.08
48	94100	-1.96	11.45	0.71	0.26	11.45	8943051756017.05	102416842186376.00	780908422906.23
49	95900	-1.94	11.47	0.67	0.24	11.47	9373039405690.19	107518709223235.00	817103072900.69
50	96500	-1.92	11.48	0.65	0.23	11.48	9519046230609.40	109252932999560.00	829380398792.99
							.		
							.		
							.		
315	252100	0.68	12.44	0.60	0.23	12.44	102496308743921.00	1274806153803640.00	8240855501664.98
316	253000	0.69	12.44	0.61	0.23	12.44	103403054391670.00	1286452369110120.00	8311377797005.24
317	253600	0.70	12.44	0.62	0.23	12.44	104010190172939.00	1294252205715250.00	8358586998762.34
318	254200	0.71	12.45	0.62	0.24	12.45	104619439614287.00	1302080639348650.00	8405951839267.64
319	255400	0.72	12.45	0.63	0.24	12.45	105844288765357.00	1317823443440790.00	8501149012035.80
320	255600	0.73	12.45	0.64	0.24	12.45	106049254475419.00	1320458399805890.00	8517075870353.51
321	256500	0.74	12.45	0.65	0.25	12.45	106974518320629.00	1332355210605200.00	8588961471267.61
322	256600	0.75	12.46	0.66	0.25	12.46	107077620391255.00	1333681072454710.00	8596970464273.46
323	263800	0.76	12.48	0.70	0.27	12.48	114656774760631.00	1431254387408420.00	9185072978056.76
324	268600	0.77	12.50	0.73	0.29	12.50	119881519370281.00	1498636303756290.00	9589770813976.04
325	269300	0.78	12.50	0.74	0.30	12.50	120655041835026.00	1508620122281220.00	9649638703081.12
326	270000	0.79	12.51	0.75	0.30	12.51	121431522648530.00	1518644144520360.00	9709723470074.48
327	270100	0.81	12.51	0.75	0.30	12.51	121542690185060.00	1520079432912290.00	9718324725385.50
328	271300	0.82	12.51	0.77	0.30	12.51	122881419296010.00	1537367040458450.00	9821885607550.86
329	271800	0.83	12.51	0.78	0.31	12.51	123441796811047.00	1544605203785910.00	9865224565209.93

330	272800	0.84	12.52	0.79	0.31	12.52	124567100695244.00	1559143393979620.00	9952235718366.43
331	273200	0.85	12.52	0.80	0.31	12.52	125018922323260.00	1564981794709290.00	9987164701665.22
332	273700	0.87	12.52	0.81	0.31	12.52	125585066962595.00	1572298401154680.00	10030926084016.30
333	275300	0.88	12.53	0.82	0.32	12.53	127406955701769.00	1595850690904230.00	10171711209395.80
334	275700	0.89	12.53	0.83	0.32	12.53	127864866058027.00	1601771948256640.00	10207085964910.20
335	276600	0.90	12.53	0.84	0.33	12.53	128898736002942.00	1615143402002280.00	10286940541972.20
336	277300	0.92	12.53	0.85	0.33	12.53	129706279914015.00	1625590028792240.00	10349300100980.70
337	279100	0.93	12.54	0.87	0.34	12.54	131796600030162.00	1652640456965110.00	10510661109803.20
338	279600	0.94	12.54	0.88	0.34	12.54	132380771594799.00	1660202518662400.00	10555741538179.10
339	280500	0.96	12.54	0.89	0.34	12.54	133436151734224.00	1673866988374420.00	10637169329045.90
340	284100	0.97	12.56	0.92	0.36	12.56	137707583831962.00	1729205362688430.00	10966527778605.90
341	286000	0.99	12.56	0.94	0.37	12.56	139994259854394.00	1758852474834630.00	11142715533332.20
342	286200	1.00	12.56	0.95	0.37	12.56	140236265174500.00	1761991001515830.00	11161356699991.00
343	290200	1.02	12.58	0.98	0.38	12.58	145128646697853.00	1825475374873130.00	11537994093080.50
344	299200	1.03	12.61	1.04	0.42	12.61	156503135932927.00	1973327308238250.00	12412148483723.80
345	299800	1.05	12.61	1.06	0.42	12.61	157279614346748.00	1983432906168890.00	12471748861342.80
346	302100	1.07	12.62	1.08	0.43	12.62	160277296481147.00	2022461208134050.00	12701757474501.30
347	302200	1.08	12.62	1.09	0.44	12.62	160408393817717.00	2024168550621950.00	12711813351350.40
348	308900	1.10	12.64	1.14	0.46	12.64	169337495234725.00	2140556817145110.00	13396134623800.90
349	310300	1.12	12.65	1.16	0.47	12.65	171239663333027.00	2165376032935600.00	13541769121114.40
350	313500	1.14	12.66	1.19	0.49	12.66	175634974839139.00	2222758020605790.00	13878093836925.00
351	318900	1.16	12.67	1.24	0.51	12.67	183202658674898.00	2321660031080120.00	14456558538390.50
352	324900	1.18	12.69	1.28	0.54	12.69	191834589249629.00	2434625089614140.00	15115472928280.10
353	334000	1.21	12.72	1.35	0.58	12.72	205379555789570.00	2612201266468530.00	16147592636820.80

354	335400	1.23	12.72	1.38	0.58	12.72	207512257967221.00	2640194878223940.00	16309908621450.60
355	346000	1.25	12.75	1.45	0.63	12.75	224086791631580.00	2858046425416370.00	17569655180258.20
356	349000	1.28	12.76	1.49	0.65	12.76	228915570867255.00	2921609874639300.00	17936117700160.40
357	369400	1.31	12.82	1.62	0.74	12.82	263388794701368.00	3376548302490820.00	20545732197304.40
358	373100	1.34	12.83	1.66	0.76	12.83	269950759046908.00	3463360733168090.00	21041242285882.70
359	380730	1.38	12.85	1.73	0.79	12.85	283786626386432.00	3646614372558490.00	22084827483222.00
360	382900	1.41	12.86	1.78	0.80	12.86	287796803921466.00	3699780198481830.00	22387005687905.00
361	440900	1.46	13.00	2.10	1.08	13.00	407656284471763.00	5298134811427580.00	31366443509682.00
362	446300	1.51	13.01	2.18	1.10	13.01	420089899076765.00	5464843067573050.00	32292880348840.40
363	457300	1.57	13.03	2.30	1.15	13.03	446107080456430.00	5814155919453810.00	34228790901097.00
364	488700	1.65	13.10	2.53	1.30	13.10	525544997669219.00	6884378842903880.00	40119457525181.40
365	523100	1.78	13.17	2.84	1.46	13.17	621583635473986.00	8184719880536030.00	47205795864543.80
TOT	63822850	-208.4	4365.0	232.58	95.28	4365.05	20913184615845900.00	259418951595684000.00	1687553151508530.00
		-0.57	11.96						

Remark: As total numbers of iterations are 343, we did not show all iterations because of space. If any one require we can provide it as we have stored as soft copy.

b_0	b_1 ($\hat{\beta}$)	Difference (b_1-b_0)	t_1 ($\hat{\theta}$)
2.3911503	2.3910504	0.0000999	4623433291804.207140

Thus, using N-R method $\hat{\beta}$ **=2.3910504**

We obtain $\hat{\theta}$ (uncensored) as $\hat{\theta}=\frac{\sum x_i^{\beta}}{n}$, on putting value of $\hat{\beta}$ we get

$\hat{\theta}$ **=4623433291804.207140.**

6.6 FINAL SOLUTION USING GOAL-SEEK FACILITY OF EXCEL

For the estimation of parameters, as we know,

$$\ln L = n\ln\beta - n\ln\theta + (\beta-1)\sum_1^n \ln x_i$$

$$\frac{\partial \ln L}{\partial \theta} = -\frac{n}{\theta} + \frac{\sum_1^n x_i^{\beta}}{\theta^2}$$

$$\frac{\partial \ln L}{\partial \beta} = \frac{n}{\beta} + \sum_1^n \ln x_i - \frac{\sum_1^n x_i^{\beta}\ln x_i}{\theta}$$

And MLE's $\hat{\beta}$ and $\hat{\theta}$ are solutions of

$$\frac{\sum_1^n x_i^{\hat{\beta}}\ln x_i}{\sum_1^n x_i^{\hat{\beta}}} - \frac{1}{\hat{\beta}} = \frac{1}{n}\sum_1^n \ln x_i = \frac{4365.045414}{365} = 11.95902853 \qquad (6.13)$$

Here, value on the left hand side of the equation is depending on the value of β. Goal seek facility of Excel is useful to find the value of β by iterative method in such a way that left hand side of the equation is equal to right hand side (11.95902853).

Goal-Seek facility of Excel

The basic forecasting command in Excel is Goal Seek, which is located on the Tools menu. The Goal Seek command determines the unknown value $\hat{\beta}$ that produces a desired result of equation (6.13).

Goal Seek is simple; it can calculate only one unknown value.

To use Goal Seek, set up the worksheet which contains the following:

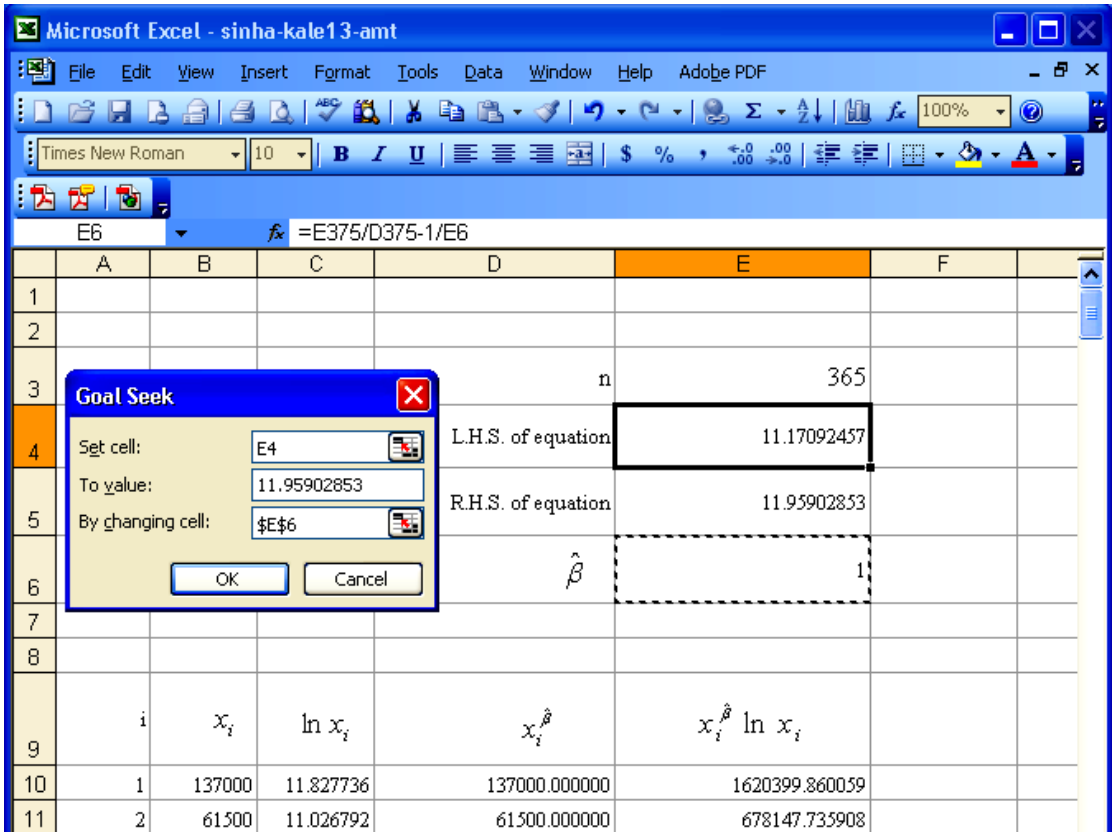
1. A formula that calculates your goal (for example, a formula that mentioned in equation-(6.13))
2. An empty variable cell for the unknown number that will produce the desired outcome (for example, a cell to hold the value of $\hat{\beta}$)
3. Values in any cells (other than the empty cell) that the formula refers to (for example, a cell that helps to evaluate the formula)

The empty cell should be referenced in your formula; it serves as the variable that Excel changes.

When the Goal Seek command starts to run, it repeatedly tries new values in the variable cell to find a solution to the problem we have set. This process is called iteration, and it continues until Excel has run the problem 100 times or has found an answer within .001 of the target value you specified. (We can adjust these iteration settings by clicking Options on the Tools menu, and then adjusting the Iteration options on the Calculations tab.) Here we have set maximum iterations 1000 and maximum change 0.0001. Because it calculates so fast, the Goal Seek command can save our significant time. To forecast using the Goal Seek command we have the following algorithm for (6.13):

1. Create a worksheet that contains clumns of i (serial number of the day), x_i (amount withdrawn per day) and ln x_i.
2. Keep one cell for the variable $\hat{\beta}$ with any initial value.
3. To calculate the left had side of the equation (6.13), add two more columns of $x_i^{\hat{\beta}}$ and $x_i^{\hat{\beta}} \ln x_i$ using the initial value of the variable.
4. The formula cell determines the value in the variable cell by following way.
    - 4.1 In your worksheet, select the cell containing the formula. (i.e. formula given in L.H.S. of equation (6.13))
    - 4.2 On the Tools menu, click Goal Seek and the Goal Seek dialog box opens.

4.3 The dialog box asks you to supply three variables, "Set *cell,* to *value,* by changing *cell*" as shown in above picture. Here we are setting Left Hand Side of the equation to the value of Right Hand Side of the equation by changing cell of variable $\hat{\beta}$. The Goal Seek command will calculate the value for this cell by using your goal in the 'To value:' text box and the formula in the cell referenced in the 'Set cell:' text box.



4.4 Click the OK button to find a solution for $\hat{\beta}$.

Excel will display the Goal Seek Status dialog box when the iteration is complete, and the result of your forecast will appear in the worksheet. Resulted worksheet showing final solution is as under:

TABLE 6.4 Final solution of $\hat{\beta}$ and $\hat{\theta}$ using goal seek of Excel

			n	365
			L.H.S. of equation	11.95910706
			R.H.S. of equation	11.95902853
			$\hat{\beta}$	2.344353817
i Day number	x_i Amount withdrawn	$\ln x_i$	$x_i^{\hat{\beta}}$	$x_i^{\hat{\beta}} \ln x_i$
1	137000	11.827736	1102276296477.690000	13037433259553.400000
2	61500	11.026792	168584505629.394000	1858946354500.900000
3	123900	11.727230	870885695573.228000	10213076914584.100000
4	196300	12.187399	2561402961634.430000	31216840867231.300000
5	140100	11.850112	1161640877782.790000	13765574194562.300000
6	152100	11.932293	1408459628465.180000	16806153639109.800000
7	109100	11.600020	646314178458.130000	7497257507448.330000
8	201200	12.212055	2713816572704.080000	33141276478327.400000
9	77300	11.255449	288152408967.913000	3243284810958.980000
10	107000	11.580584	617525727915.382000	7151308634339.850000
11	143000	11.870600	1218797956106.680000	14467862907144.400000
12	151400	11.927681	1393310364972.950000	16618961037912.700000
13	195000	12.180755	2521812732407.910000	30717582639662.400000
14	35800	10.485703	47414705299.444700	497176525776.267000
15	117500	11.674194	769064252127.986000	8978204979845.650000
16	17000	9.740969	8273115288.009010	80588156435.274700
17	183100	12.117788	2175722522903.350000	26364943693364.600000
18	108100	11.590812	632511578058.684000	7331322791395.840000
19	103500	11.547327	571208354707.685000	6595929595072.660000
20	95900	11.471061	477689249197.555000	5479602641204.880000
21	53700	10.891168	122668561061.524000	1336003941247.470000
22	101300	11.525842	543149781978.534000	6260258401171.200000
23	37100	10.521372	51550058945.522200	542377359602.881000
24	137400	11.830652	1109836001132.560000	13130083127762.000000
25	183700	12.121059	2192473705733.660000	26575103737752.300000
26	108400	11.593583	636634421532.110000	7380874240963.130000
27	158900	11.976030	1560539123104.560000	18689063904602.700000
28	106000	11.571194	604080736464.402000	6989935618671.490000
29	163600	12.005180	1670908924762.960000	20059561909377.800000
30	53900	10.894886	123742299445.066000	1348158215749.750000
31	256500	12.454884	4795287176125.570000	59724745254843.500000

32	189800	12.153726	2366976044585.970000	28767578685386.600000
33	126600	11.748788	916030499367.671000	10762247945040.500000
34	153600	11.942107	1441239123556.600000	17211431969784.400000
35	135000	11.813030	1064921395393.690000	12579948452575.900000
36	145500	11.887931	1269338827968.010000	15089812866566.800000
37	74800	11.222573	266777878473.495000	2993934259695.510000
38	217800	12.291332	3268107664627.560000	40169397917358.500000
39	105100	11.562668	592125132178.557000	6846546055445.420000
40	164400	12.010058	1690126941874.580000	20298522196357.200000
41	188000	12.144197	2314686002066.760000	28110003361960.200000
42	132800	11.796600	1024681658069.920000	12087759151666.800000
43	67200	11.115429	207520775294.185000	2306682345548.710000
44	83500	11.332602	345282426285.601000	3912948283903.310000
45	108600	11.595427	639391520591.758000	7414017500979.960000
46	106200	11.573079	606756161841.632000	7022037230023.960000
47	195200	12.181780	2527880524822.670000	30794084300765.600000
48	85800	11.359774	367993166701.581000	4180319312327.520000
49	90900	11.417515	421335610439.186000	4810605770267.290000
50	243800	12.404103	4257091843074.640000	52805407813600.000000

.
.
.

315	253000	12.441145	4643294370079.130000	57767897457245.600000
316	318900	12.672633	7989387385578.450000	101246573057810.000000
317	120900	11.702719	822253059186.332000	9622596528644.840000
318	276600	12.530328	5723041823671.540000	71711589481181.700000
319	201800	12.215032	2732827219472.590000	33381572993664.100000
320	225300	12.325188	3538067638414.670000	43607349247460.500000
321	277300	12.532855	5757053990880.590000	72152324228233.800000
322	346000	12.754194	9672844246360.940000	123369332572590.000000
323	214000	12.273731	3135998711980.580000	38490405529192.200000
324	167900	12.031124	1775691612944.080000	21363565702423.200000
325	199700	12.204572	2666622552873.230000	32544985661796.800000
326	140500	11.852963	1169431099943.630000	13861223287087.700000
327	193500	12.173033	2476570581006.550000	30147374892938.000000
328	188800	12.148444	2337843329483.440000	28401157676513.800000
329	153500	11.941456	1439040363173.760000	17184236957464.700000
330	201300	12.212552	2716979729688.680000	33181255176569.700000
331	83200	11.329003	342381192324.663000	3878837427216.210000
332	215800	12.282107	3198187003875.630000	39280476048788.100000
333	286200	12.564446	6199608624279.970000	77894648686690.600000

334	286000	12.563747	6189456807601.790000	77762769953962.100000
335	113400	11.638677	707622132188.739000	8235785201275.880000
336	51600	10.851277	111716755952.054000	1212269458955.480000
337	136800	11.826275	1098507551489.650000	12991252705656.700000
338	145700	11.889305	1273433026242.700000	15140233636118.500000
339	99400	11.506907	519567342009.692000	5978613288748.070000
340	176100	12.078807	1985710668522.760000	23985016507688.000000
341	132100	11.791314	1012064209648.430000	11933567380554.500000
342	127100	11.752729	924534461476.315000	10865803399568.500000
343	107300	11.583384	621592348532.011000	7200142820138.100000
344	169200	12.038837	1808091221453.600000	21767315001072.700000
345	168800	12.036470	1798086323999.120000	21642611846550.600000
346	133800	11.804101	1042862274384.520000	12310052060889.200000
347	275300	12.525617	5660182774246.180000	70897279831948.400000
348	237600	12.378344	4007616098284.950000	49607650149374.500000
349	227100	12.333146	3604691340037.490000	44457183601479.600000
350	117000	11.669929	761414021559.062000	8885647733973.730000
351	225700	12.326962	3552811327614.270000	43795370096526.000000
352	137000	11.827736	1102276296477.690000	13037433259553.400000
353	214200	12.274665	3142873954525.890000	38577726302453.200000
354	172000	12.055250	1879018683548.990000	22652039525989.300000
355	161100	11.989781	1611663259477.720000	19323488832523.200000
356	196800	12.189943	2576724201549.960000	31410121822839.800000
357	96700	11.479369	487083685886.087000	5591413209001.780000
358	217000	12.287653	3240035332577.150000	39812428683807.900000
359	136000	11.820410	1083506533539.940000	12807491642594.100000
360	197700	12.194506	2604434561811.690000	31759792914473.300000
361	382900	12.855529	12266692798261.700000	157694826688390.000000
362	373100	12.829602	11543294881079.700000	148095876313371.000000
363	279100	12.539325	5845044999513.540000	73292921340664.200000
364	221900	12.309982	3414163205780.410000	42028287980092.100000
365	279600	12.541115	5869622852860.950000	73611616902684.500000
Total	63822850	4365.045414	945077923974696.000000	11705417466748100.000000
		Theta	2589254586232.040000	

Now, $\hat{\theta} = \frac{\sum_{1}^{n} x_i^{\hat{\beta}}}{n}$. Which is calculated from the final value as shown above.

$$= \frac{945077923974696.000000}{365} = \mathbf{2589254586232.040000}$$

TABLE 6.5 Final solutions using all the methods

	Method of calculation		
	Graphical	**Newton-Raphson**	**Goal Seek of Excel**
$\hat{\beta}$	2.4410138	2.3910504	2.344353817
$\hat{\theta}$	8578516936201.362400	4623433291804.207140	2589254586232.040000

6.7 GOODNESS OF FIT TEST

Using Excel we have prepared a worksheet 'goodness-amt.xls' having columns like x and f(x) for unique values of x with the help of $\hat{\beta}$=2.3910504 and $\hat{\theta}$=4623433291804.207140 (obtained using N-R method).

Where $f(x,\beta,\theta) = \frac{\beta}{\theta} x^{\beta-1} \exp\left(-\frac{x^{\beta}}{\theta}\right)$

$$F(x) = 1 - e^{-\frac{x^{\beta}}{\theta}}$$

$$\int_{b}^{a} f(x)dx = F(b) - F(a)$$

$$= e^{-\frac{a^{\hat{\beta}}}{\theta}} - e^{-\frac{b^{\hat{\beta}}}{\theta}}$$

And $\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$

TABLE 6.6

(a) Lower Limit	(b) Upper Limit	Oi Observed frequency	Weibull Prob.	Ei Exp. Freq.= 365*W.Prob	Oi (Merged)	Ei (Merged)	$\frac{(O_i - E_i)^2}{E_i}$
0	25000	3	0.0071	2.5779			
25000	50000	7	0.0294	10.7425	10	13.3204	0.8277
50000	75000	17	0.0569	20.7588	17	20.7588	0.6806
75000	100000	34	0.0838	30.5863	34	30.5863	0.3810
100000	125000	39	0.1057	38.5737	39	38.5737	0.0047
125000	150000	46	0.1191	43.4825	46	43.4825	0.1458
150000	175000	44	0.1225	44.6982	44	44.6982	0.0109
175000	200000	52	0.1160	42.3333	52	42.3333	2.2073
200000	225000	45	0.1018	37.1444	45	37.1444	1.6614
225000	250000	24	0.0830	30.2839	24	30.2839	1.3039
250000	275000	21	0.0629	22.9763	21	22.9763	0.1700
275000	300000	13	0.0445	16.2311	13	16.2311	0.6432
300000	325000	7	0.0292	10.6761	7	10.6761	1.2658
325000	350000	4	0.0179	6.5359	13	13.9225	0.0611
350000	375000	2	0.0102	3.7219			
375000	400000	2	0.0054	1.9698			
400000	425000	0	0.0027	0.9680			
425000	450000	2	0.0012	0.4413			
450000	475000	1	0.0005	0.1864			
475000	500000	1	0.0002	0.0729			
500000	525000	1	0.0001	0.0264			
		365	1.0000	364.9875	365	364.9875	**9.3634**

Null hypothesis H_0: Amount withdrawn per day fits the Weibull distribution.

Alternate hypothesis H_1: Amount withdrawn per day does not fit the Weibull distribution.

The calculated value of χ^2 is 9.3634 and tabulated value of $\chi^2_{0.01}$ for 12 d.f. is 26.217 and $\chi^2_{0.005}$ for 12 d.f. is 28.300. Since the calculated value is much less than the tabulated value, it is insignificant and hence we accept the null hypothesis both at 1% and 0.5% level of significance. In other words we can say that amount withdrawn from the ATM per day of a bank at Porbandar fits the Weibull distribution.

6.8 MEAN AND VARIANCE

Now we find expected mean and variance of Weibull distribution with the help of 'mean-weibull.xls' file of Excel, for the following value of $\hat{\beta}$ and $\hat{\theta}$ (obtained from different methods)

	$\hat{\beta}$	$\hat{\theta}$
Graphical	2.4410138	8578516936201.360
N-R	2.3910504	4623433291804.200
Goal Seek	2.344535817	2589254586232.040

Mean:

Here mean is obtained using the formula $\mu_1' = \theta^{\frac{1}{\beta}}\Gamma\left(\frac{\beta+1}{\beta}\right)$ for the different methods as:

TABLE 6.7

Method	$\theta^{\frac{1}{\beta}}$	$\left(\frac{\beta+1}{\beta}\right)$	$\Gamma\left(\frac{\beta+1}{\beta}\right)$	Mean
Graphical	198782.13165	1.409665853	0.8867799	176275.993487
N-R	198070.78600	1.418226232	0.8864218	175574.256652
Goal Seek	197021.10056	1.426523661	0.8861373	174587.737295

Variance:

Similarly variance is obtained using the formula $\mu_2 = \theta^{\frac{2}{\beta}}\left[\Gamma\left(\frac{\beta+2}{\beta}\right) - \Gamma^2\left(\frac{\beta+1}{\beta}\right)\right]$ for different methods as:

TABLE 6.8

Method	$\theta^{\frac{2}{\beta}}$	$\left(\frac{\beta+2}{\beta}\right)$	$\left(\frac{\beta+1}{\beta}\right)$	$\Gamma\left(\frac{\beta+2}{\beta}\right)$	$\Gamma^2\left(\frac{\beta+1}{\beta}\right)$	**Variance**
Graphical	39514335863	1.81933	1.40966	0.93665	0.78637	**5938179522**
N-R	39232036266	1.83645	1.41822	0.94156	0.78574	**6113271086**
Goal Seek	38817314067	1.85304	1.42652	0.94654	0.78523	**6261274822**

Conclusion: From all the three methods, we concluded that withrawl of average amount per day is more or less same. We also found from the observed amount withdrawn per day with the help of available data, that the average withdrawn amount is Rs.173017 only.

Finally we conclude that the Bank must keep more than Rs.176275 per day at the ATM so that customer will not return without withdrawing money. However during the festival period, Bank must keep more amount which must be equal to its mode amount.

References

1. *http://www.en.wikipedia.org/wiki/Time_series*
2. Ajay Goel & Alka Goel, Mathematics and Statistics, Taxmann Allied Services Pvt. Ltd., 879-929
3. *http://www.ecompare.co.in,* Loans, Home Loans & Rates, Personal Loans, Credit Cards in India
4. *http://www.ecompare.co.in/savings/,* Fixed Deposits, Interest Rates ...
5. *http://www.rbi.org.in/*, Reserve Bank of India
6. *http://www.en.wikipedia.org/wiki/Deposit_account*
7. *http://www.allbankingsolutions.com/dep1.htm,* Saving, Recurring, Term, Fixed *Deposits* in Banks in *India*. Tips ...
8. *http://www.en.wikipedia.org/wiki/Weibull_distribution*
9. Johnson and Kotz, Continuous univariate distributions-1, John Wiley & Sons, 250-267
10. Sinha & Kale, Life Testing and Reliability Estimation, Wiley Eastern Limited, 41-45

Annexure-I

C++ Programs

1. daytot.cpp

```
#include <stdio.h>
#include <conio.h>
main()
{
     FILE *fp1,*fp2;
     struct atm
     {
          int sr;
          int yy;
          int mm;
          int dd;
          long int dt;
          double tm;
          double amt;
     };
     struct atm day;
     int tmpdd, tmpmm,tmpyy,tothits,gtothits=0;
     double totamt, avgamt;
     int i,j,dayno;
     clrscr();
     fp1=fopen("ymdtamt.csv","r");
     fp2=fopen("dailytot.txt","w");

     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     dayno=1;
     tothits=0;
     totamt=0;
     while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
          &day.yy,&day.mm,&day.dd,&day.dt,&day.tm,
          &day.amt)!=EOF)
          {
          tmpdd=day.dd;
```

```
            tmpmm=day.mm;
            tmpyy=day.yy;
            tothits=tothits+1;
            totamt=totamt+day.amt;
            while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
                 &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
                 &day.tm,&day.amt)!=EOF && tmpdd==day.dd)
                 {
                 totamt=totamt+day.amt;
                 tothits=tothits+1;
                 }
            avgamt=totamt/tothits;
            gtothits=gtothits+tothits;
            fprintf(fp2,"%d,%d,%d,%d,%d,%lf,%lf\n",dayno,
            tmpdd,tmpmm,tmpyy,tothits,totamt, avgamt);
            tothits=1;
            totamt=day.amt;
            dayno++;
            }
      fclose(fp1);
      fclose(fp2);
      printf("Grand total of hits=%d\n",gtothits);
      printf("file reading and creation over...");
      getch();
}
```

2. mmlytot.cpp

```
#include <stdio.h>
#include <conio.h>
main()
{
      FILE *fp1,*fp2;
      struct atm
      {
            int sr;
            int yy;
            int mm;
            int dd;
            long int dt;
            double tm;
            double amt;
      };
      struct atm day;
      int tmpmm, tothits, gtothits;
      double totamt, avgamt,gtotamt;
      int i,j;
      clrscr();
      fp1=fopen("ymdtamt.csv","r");
      fp2=fopen("mmlytot.txt","w");
      if(fp1==NULL)
```

```
        {
             puts("cannot open the file");
        }
        printf("\n\n ----------------------------------------
        ------\n");
        printf(" Month    Total no.   Total
            Average\n");
        printf(" Number   of hits     Amount(Rs.)
        Amount(Rs.)\n");
        printf(" ----------------------------------------
        ------\n");
        tothits=0;
        totamt=0;
        gtothits=0;
        gtotamt=0;
        while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
             &day.yy,&day.mm,&day.dd,&day.dt,&day.tm,
             &day.amt)!=EOF)
             {
             tmpmm=day.mm;
             tothits=tothits+1;
             totamt=totamt+day.amt;
             while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
                  &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
                  &day.tm,&day.amt)!=EOF && tmpmm==day.mm)
                  {
                  totamt=totamt+day.amt;
                  tothits=tothits+1;
                  }
             avgamt=totamt/tothits;
             gtothits=gtothits+tothits;
             gtotamt=gtotamt+totamt;
             fprintf(fp2,"\n%d,%d,%lf,%lf",tmpmm,tothits,
             totamt, avgamt);
             printf("\n  %2d     %7d   12.2lf%11.2lf",tmpmm,
             tothits, totamt, avgamt);
             tothits=1;
             totamt=day.amt;
             }
        printf("\n ----------------------------------------
        ----\n");
        printf("         %7d    %12.2lf\n",
        gtothits,gtotamt);
        printf(" ----------------------------------------
        --\n");
        fclose(fp1);
        fclose(fp2);
        printf(" file reading and creation over...");
        getch();
}
```

3. qmahit.cpp

```
#include <stdio.h>
# include <conio.h>
main()
{
     FILE *fp1,*fp2;
     struct atm
     {
          int mm;
          int tothits;
          double totamt;
          double avgamt;
     };
     struct atm day;
     int mnth[13], hits[13], mtot3[13]={0}, i;
     float mavg3[13];
     clrscr();
     fp1=fopen("mmlytot.txt","r");
     fp2=fopen("qmahit.txt","w");
     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     i=1;
     while(fscanf(fp1,"%d,%d,%lf,%lf",&day.mm,
          &day.tothits,&day.totamt,&day.avgamt)!=EOF)
          {
          mnth[i]=day.mm;
          hits[i]=day.tothits;
          i++;
          }
     fclose(fp1);
     for(i=2;i<=11;i++)
          {
          mtot3[i]=hits[i-1]+hits[i]+hits[i+1];
          mavg3[i]=mtot3[i]/3;
          }
          printf("    Quarterly moving average for
           hits\n");
          printf("----------------------------------------
          -----\n");
          printf(" Month       Hits    Quarterly
             Quarterly\n");
          printf("Number                   moving
                 moving\n");
          printf("                          total
             average\n");
          printf("----------------------------------------
          -----\n");
     for(i=1;i<=12;i++)
```

```
            {
            if(i==1||i==12)
                  {
                  printf("  %2d        %5d
                  %5d\n",mnth[i],hits[i],mtot3[i]);
                  continue;
                  }
            printf("  %2d        %5d        %5d   %8.2f\n",
            mnth[i],hits[i],mtot3[i],mavg3[i]);
            fprintf(fp2,"%2d,%8.2f\n",mnth[i],mavg3[i]);
            }
      printf("---------------------------------------\n");
      fclose(fp2);
      getch();
}
```

4. qmaamt.cpp

```
#include <stdio.h>
# include <conio.h>
main()
{
      FILE *fp1,*fp2;
      struct atm
      {
            int mm;
            int tothits;
            double totamt;
            double avgamt;
      };
      struct atm day;
      int mnth[13],i;
      double amt[13], mtot3[13]={0}, mavg3[13]={0};
      clrscr();
      fp1=fopen("mmlytot.txt","r");
      fp2=fopen("qmaamt.txt","w");
      if(fp1==NULL)
      {
            puts("cannot open the file");
      }
      i=1;
      while(fscanf(fp1,"%d,%d,%lf,%lf",&day.mm,
      &day.tothits,&day.totamt,&day.avgamt)!=EOF)
            {
            mnth[i]=day.mm;
            amt[i]=day.totamt;
            i++;
            }
      fclose(fp1);
      for(i=2;i<=11;i++)
            {
```

```
            mtot3[i]=amt[i-1]+amt[i]+amt[i+1];
            mavg3[i]=mtot3[i]/3;
            }
      printf("   Quarterly moving average for amount\n");
      printf("-----------------------------------------
      ----------\n");
      printf(" Month    Transaction        Quarterly
            Quarterly\n");
      printf("Number      of amount           moving
              moving\n");
      printf("         During month            total
              average\n");
      printf("-----------------------------------------
      ----------\n");
      for(i=1;i<=12;i++)
            {
            if(i==1 || i==12)
                  {
                  printf("    %2d%15.2lf  %15.2lf\n",
                  mnth[i],amt[i],mtot3[i]);
                  continue;
                  }
            printf("    %2d%15.2lf  %15.2lf    %15.2lf\n",
            mnth[i],amt[i],mtot3[i],mavg3[i]);
            fprintf(fp2,"%2d,%15.2lf\n",mnth[i],mavg3[i]);
            }
      printf("-----------------------------------------
      ----------\n");
      fclose(fp2);
      getch();
}
```

5. irrvartn.cpp

```
#include <stdio.h>
# include <conio.h>
main()
{
      FILE *fp1,*fp2,*fp3;
      struct atm
      {
            int sr;
            int yy;
            int mm;
            int dd;
            long int dt;
            double tm;
            double amt;
      };
      struct atm day;
      int tmpmm;
```

```
int i,j,k,hit2[13][5]={{0},{0}};
int mtothit[13]={0},sumhit=0;
float mavghit[13]={0},xdev[13],xdev2[13],xdevy[13];
float trend[13];
float weektr[13][5],perc[13][5],totperc[13]={0};
float avgperc[13],savgperc, adjsidx[13];
float cycirr[13],mtot3[13]={0},mavg3[13]={0};
float irrvar[13]={0};
float sumy=0,sumxdevy=0,sumx=0,sumxdev=0,sumxdev2=0;
float xbar,a,b,weekincr;
clrscr();
fp1=fopen("ymdtamt.csv","r");
fp2=fopen("irrvar.txt","w");
fp3=fopen("leastsq.txt","w");
if(fp1==NULL)
{
     puts("cannot open the file");
}
i=1;
while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
     &day.yy,&day.mm,&day.dd,&day.dt,&day.tm,
     &day.amt)!=EOF)
     {
     tmpmm=day.mm;
     if(day.dd>=1 && day.dd<=7)
          j=1;
     if(day.dd>=8 && day.dd<=14)
          j=2;
     if(day.dd>=15 && day.dd<=21)
          j=3;
     if(day.dd>=22 && day.dd<=31)
          j=4;
     i=tmpmm;
     hit2[i][j]++;
while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
&day.yy,&day.mm,&day.dd,&day.dt,&day.tm,&day.amt)
!=EOF && tmpmm==day.mm)
          {
          if(day.dd>=1 && day.dd<=7)
               j=1;
          if(day.dd>=8 && day.dd<=14)
               j=2;
          if(day.dd>=15 && day.dd<=21)
               j=3;
          if(day.dd>=22 && day.dd<=31)
               j=4;
          i=tmpmm;
          hit2[i][j]++;
          }
     tmpmm=day.mm;
     if(day.dd>=1 && day.dd<=7)
```

```
          j=1;
     if(day.dd>=8 && day.dd<=14)
          j=2;
     if(day.dd>=15 && day.dd<=21)
          j=3;
     if(day.dd>=22 && day.dd<=31)
          j=4;
     i=tmpmm;
     hit2[i][j]++;
     }
hit2[12][4]--;
fclose(fp1);
for(i=1;i<=12;i++)
     {
     for(j=1;j<=4;j++)
          {
          mtothit[i]=mtothit[i]+hit2[i][j];
          }
     mavghit[i]=mtothit[i]/4.0;
     sumhit=sumhit+mtothit[i];
     sumx=sumx+i;
     }
xbar=sumx/12.0;
for(i=1;i<=12;i++)
     {
     xdev[i]=i-xbar;
     xdev2[i]=xdev[i]*xdev[i];
     xdevy[i]=xdev[i]*mavghit[i];
     sumxdev=sumxdev+xdev[i];
     sumy=sumy+mavghit[i];
     sumxdev2=sumxdev2+xdev2[i];
     sumxdevy=sumxdevy+xdevy[i];
     }
a=sumy/12.0;
b=sumxdevy/sumxdev2;
for(i=1;i<=12;i++)
     trend[i]=a+b*xdev[i];
printf("\n\n  Table-1 : Showing weekly total number
     of hits\n\n");
printf(" ------------------------------------------
     ----------------------------------\n");
printf("                          WEEK           \n");
printf(" Month        ----------------------
    Average                       TrendY\n");
printf("     X   Xdev     I    II   III  Last
   Total       Y  Xdev^2    XdevY a+b*xdev\n");
printf(" ------------------------------------------
     ----------------------------------\n");
for(i=1;i<=12;i++)
     {
     printf(" %5d%7.2f",i,xdev[i]);
```

```
          for(j=1;j<=4;j++)
               {
               printf("%6d",hit2[i][j]);
               }
          printf("  6d%8.2f%8.2f%9.2f%9.2f\n",mtothit[i],
               mavghit[i], xdev2[i], xdevy[i], trend[i]);
          fprintf(fp3,"%d,%9.2f\n",i,trend[i]);
          }
     printf(" ----------------------------------------
          ----------------------------------\n");
     printf("      %7.2f                          %6d%8.2f
          %7.2f%9.2f\n",sumxdev,sumhit,sumy,sumxdev2,
          sumxdevy);
     printf(" ----------------------------------------
          ----------------------------------\n");
     printf("\n a=%-8.2f b=%-8.2f\n",a,b);
     printf("\n Press any key to continue....\n");
     fclose(fp3);
     getch();
     clrscr();
/*
Here x is measured in months and the origin is last week
of June or the first week of july. From this equation
value of y increase by 17.62 every month or by
17.62/4=4.405 every week. Thus in 1st week of July the
value of Y is 370.85, 2nd week 370.85+4.405 and so on
continuously adding/subtracting 4.405 we will get
different values of Y.
*/
     weekincr=b/4.0;
     k=0;
     for(i=7;i<=12;i++)
          for(j=1;j<=4;j++)
               {
               weektr[i][j]=hit2[7][1]+k*weekincr;
               k++;
               }
     k=1;
     for(i=6;i>=1;i--)
          for(j=4;j>=1;j--)
               {
               weektr[i][j]=hit2[7][1]-k*weekincr;
               k++;
               }
     printf("\n\n Table-2 : Showing weekly trend of
     hits\n\n");
     printf(" ---------------------------------------\n");
     printf("                      WEEK               \n");
     printf(" Month      I      II     III    Last  \n");
     printf(" ---------------------------------------\n");
     for(i=1;i<=12;i++)
```

```
          {
          printf("%5d",i);
          for(j=1;j<=4;j++)
               {
               printf("%8.2f",weektr[i][j]);
               }
          printf("\n");
          }
     printf(" ----------------------------------------
     \n");
     printf(" Press any key to continue...\n");
     getch();
     clrscr();
/*
we now devide each of the given weekly values in the
first table by
the corresponding trend values in second table. The
results expressed
as percentages are shown in following table. The first
entry of the
table is 241/281.28*100
*/
     savgperc=0;
     for(i=1;i<=12;i++)
          {
          for(j=1;j<=4;j++)
               {
               perc[i][j]=hit2[i][j]/weektr[i][j]*100.00;
               totperc[i]+=perc[i][j];
               }
          avgperc[i]=totperc[i]/4.0;
          savgperc=savgperc+avgperc[i];
          }
     printf("\n\n       Table Showing weekly percentages
     of hits\n\n");
     printf(" ----------------------------------------
     ---------------------------\n");
     printf("                          WEEK
           Total                  Adjusted\n");
     printf(" Month       I      II     III    Last
      Perc   Average  Seasonal Index\n");
     printf(" ----------------------------------------
     ---------------------------\n");
     for(i=1;i<=12;i++)
          {
          printf("%5d ",i);
          for(j=1;j<=4;j++)
               {
               printf("%8.2f",perc[i][j]);
               }
          adjsidx[i]=avgperc[i]*1200/savgperc;
```

```
          printf("%8.2f %8.2f        %8.2f\n",totperc[i],
          avgperc[i], adjsidx[i]);
          }
     printf(" ------------------------------------------
     ----------------------------\n");
     printf("                          %9.2f\n",savgperc);
     printf(" ------------------------------------------
     ----------------------------\n");
     printf(" Press any key to continue...\n");
     getch();
     clrscr();
/*   To calculate Cyclical Irregulars
     Take Observed values Yt=Monthly average hit avghit[]
          Trend values Tt = trend[]
          Seasonal index = Adjusted seasional index =
          adjsidx[]
          Cyclical Irregulars = cycirr[]
*/
     for(i=1;i<=12;i++)
          {
          cycirr[i]=(mavghit[i]*100)/((trend[i]*
          adjsidx[i])/100.00);
          }
     for(i=2;i<=11;i++)
          {
          mtot3[i]=cycirr[i-1]+cycirr[i]+cycirr[i+1];
          mavg3[i]=mtot3[i]/3.0;
          irrvar[i]=cycirr[i]/mavg3[i]*100;
          }

     printf("\n\n            Table-4 : Showing Cyclical
     regularities\n\n");
     printf(" ------------------------------------------
     ----------------------------\n");
     printf(" Month Observed    Trend Seasonal  Cyclical
        Three     Three   Irragular\n");
     printf("         Values   Values    Index  Irrelars
          Monthly   Monthly  Variations\n");
     printf("                                  (%)    Moving
       Moving         (%)\n");
     printf("                                        Totals
          Averages     CtRt/Ct\n");
     printf("     X       Yt       Tt              CtRt
            (%)  Ct          Rt\n");
     printf("     I       II      III       IV         V
      VI       VII        VIII\n");
     printf(" ------------------------------------------
          ----------------------------\n");
```

```
     for(i=1;i<=12;i++)
          {
          printf(" %5d%9.2f%9.2f%9.2f %9.2f%9.2f %9.2f
               %10.4f\n",i, mavghit[i], trend[i],
               adjsidx[i],cycirr[i],mtot3[i], mavg3[i],
               irrvar[i]);
          fprintf(fp2,"%d,%f,%f,%f,%f,%f,%f\n",i,
               mavghit[i],trend[i],adjsidx[i],cycirr[i],
               mavg3[i],irrvar[i]);
          }
     printf(" ------------------------------------------
          ----------------------------\n");
     printf(" Press any key to continue...\n");
     getch();
     clrscr();
}
```

6. irramt.cpp

```
/*
Monthly, Weekly number of hits for time series
1st week- Date 1 to 7
2nd week= Date 8 to 14
3rd week- Date 16 to 21
Last week- Date 22 to end of the month
Program to calculate monthly trend using least square
method*/

#include <stdio.h>
# include <conio.h>
main()
{
     FILE *fp1,*fp2,*fp3;
     struct atm
     {
          int sr;
          int yy;
          int mm;
          int dd;
          long int dt;
          double tm;
          double amt;
     };
     struct atm day;
     int tmpmm;
     int i,j,k;
     double amt2[13][5]={{0},{0}},mtotamt[13]={0};
     double mavgamt[13]={0},xdev[13],xdev2[13];
     double xdevy[13],trend[13], irrvar[13]={0};
     double weektr[13][5],perc[13][5],totperc[13]={0};
     double avgperc[13],savgperc, adjsidx[13];
```

```
double cycirr[13],mtot3[13]={0},mavg3[13]={0},;
double sumy=0,sumxdevy=0,sumx=0,sumxdev=0;
double sumxdev2=0,xbar,a,b,weekincr;
clrscr();
fp1=fopen("ymdtamt.csv","r");
fp2=fopen("irramt.txt","w");
fp3=fopen("leastamt.txt","w");
if(fp1==NULL)
     {
     puts("cannot open the file");
     }
i=1;
while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
     &day.yy,&day.mm,&day.dd,&day.dt,&day.tm,
     &day.amt)!=EOF)
     {
     tmpmm=day.mm;
     while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
          &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
          &day.tm,&day.amt)!=EOF && tmpmm==day.mm)
          {
          if(day.dd>=1 && day.dd<=7)
               j=1;
          if(day.dd>=8 && day.dd<=14)
               j=2;
          if(day.dd>=15 && day.dd<=21)
               j=3;
          if(day.dd>=22 && day.dd<=31)
               j=4;
          i=tmpmm;
          amt2[i][j]=amt2[i][j]+day.amt;
          }
     }
fclose(fp1);
for(i=1;i<=12;i++)
     {
     for(j=1;j<=4;j++)
          {
          mtotamt[i]=mtotamt[i]+amt2[i][j];
          }
     mavgamt[i]=mtotamt[i]/4.0;
     sumx=sumx+i;
     }
xbar=sumx/12.0;
for(i=1;i<=12;i++)
     {
     xdev[i]=i-xbar;
     xdev2[i]=xdev[i]*xdev[i];
     xdevy[i]=xdev[i]*mavgamt[i];
     sumxdev=sumxdev+xdev[i];
     sumy=sumy+mavgamt[i];
```

```
     sumxdev2=sumxdev2+xdev2[i];
     sumxdevy=sumxdevy+xdevy[i];
     }
a=sumy/12.0;
b=sumxdevy/sumxdev2;
for(i=1;i<=12;i++)
     trend[i]=a+b*xdev[i];
printf("\n      weekly total amount withdrawn\n\n");
printf(" ------------------------------------------
     --------\n");
printf(" Month                      WEEK       \n");
printf("              ------------------------------
     \n");
printf("     X  Xdev       I      II     III    Last
   Total\n");
printf(" ------------------------------------------
     --------\n");
for(i=1;i<=12;i++)
     {
     printf(" %5d%6.2lf",i,xdev[i]);
     for(j=1;j<=4;j++)
          {
          printf("%8.0lf",amt2[i][j]);
          }
     printf("%8.0lf\n",mtotamt[i]);
     }
printf(" ------------------------------------------
     --------\n");
printf("      %6.2lf\n",sumxdev);
printf(" ------------------------------------------
     --------\n");
printf(" Press any key to continue....\n");
getch();
clrscr();
printf("\n\n Least square method for weekly total
     amount    withdrawn\n\n");
printf(" ------------------------------------------
     -------------------------\n");
printf("                                  Weekly\n");
printf(" Month            Weekly      Average
       TrendY\n");
printf("     X  Xdev       Total            Y
     Xdev^2        XdevY
a+b*xdev\n");
printf(" ------------------------------------------
     -------------------------\n");
for(i=1;i<=12;i++)
     {
     printf(" %5d%6.2lf",i,xdev[i]);
     printf(" %12.2lf%12.2lf%8.2lf%13.2lf%12.2lf\n",
          mtotamt[i mavgamt[i],xdev2[i],xdevy[i],
```

```
          trend[i]);
     fprintf(fp3,"%d,%12.2lf\n",i,trend[i]);
     }
printf(" ------------------------------------------
     -------------------------\n");
printf("      %6.2lf        %12.2lf%8.2lf%13.2lf\n",
     sumxdev,sumy,sumxdev2,sumxdevy);
printf(" ------------------------------------------
     -------------------------\n");
printf("\n a=%-12.2lf b=%-12.2lf\n",a,b);
printf("\n Press any key to continue....\n");
fclose(fp3);
getch();
clrscr();
weekincr=b/4.0;
k=0;
for(i=7;i<=12;i++)
     for(j=1;j<=4;j++)
          {
          weektr[i][j]=amt2[7][1]+k*weekincr;
          k++;
          }
k=1;
for(i=6;i>=1;i--)
     for(j=4;j>=1;j--)
          {
          weektr[i][j]=amt2[7][1]-k*weekincr;
          k++;
          }
printf("  Table showing weekly trend of amount
     withdrawn\n\n");
printf("------------------------------------------
     ---------\n");
printf("                                WEEK\n");
printf("Month           I          II         III
       Last \n");
printf("------------------------------------------
     ---------\n");
for(i=1;i<=12;i++)
     {
     printf("%5d",i);
     for(j=1;j<=4;j++)
          {
          printf("%12.2lf",weektr[i][j]);
          }
     printf("\n");
     }
printf("------------------------------------------
     ---------\n");
printf("Press any key to continue...\n");
getch();
```

```
     clrscr();
/*   we now devide each of the given weekly values in the
     first table by the corresponding trend values in
     second table. The results expressed as percentages
     are shown in following table. The first entry of the
     table is 241/281.28*100
*/
     savgperc=0;
     for(i=1;i<=12;i++)
          {
          for(j=1;j<=4;j++)
               {
               perc[i][j]=amt2[i][j]/weektr[i][j]*100.00;
               totperc[i]+=perc[i][j];
               }
          avgperc[i]=totperc[i]/4.0;
          savgperc=savgperc+avgperc[i];
          }
     printf("  Table showing weekly percentages of amount
      withdrawn\n\n");
     printf("-----------------------------------------
          ---------------------------\n");
     printf("                        WEEK             Total
                Adjusted\n");
     printf("Month       I      II     III    Last
          Perc   Average      Seasonal Index\n");
     printf("-----------------------------------------
          ---------------------------\n");
     for(i=1;i<=12;i++)
          {
          printf("%5d",i);
          for(j=1;j<=4;j++)
               {
               printf("%8.2lf",perc[i][j]);
               }
          adjsidx[i]=avgperc[i]*1200/savgperc;
          printf("%8.2lf  %8.2lf    %8.2lf\n",totperc[i],
               avgperc[i],adjsidx[i]);
          }
     printf("-----------------------------------------
          ---------------------------\n");
     printf("                         %9.2lf\n",savgperc);
     printf("-----------------------------------------
          ---------------------------\n");
     printf("Press any key to continue...\n");
     getch();
     clrscr();
/*   To calculate Cyclical Irregulars
     Take Observed values Yt=
     Monthly average hit mavghit[]
     Trend values Tt = trend[]
```

```
     Seasonal index = Average percentage = avgperc[]
     Cyclical Irregulars = cycirr[]
*/
     for(i=1;i<=12;i++)
          {
          cycirr[i]=(mavgamt[i]*100)/((trend[i]*
          adjsidx[i])/100.00);
          }
     for(i=2;i<=11;i++)
          {
          mtot3[i]=cycirr[i-1]+cycirr[i]+cycirr[i+1];
          mavg3[i]=mtot3[i]/3.0;
          irrvar[i]=cycirr[i]/mavg3[i]*100;
          }

     printf("       Table showing Cyclical Irregularities
          in amount withdrawn\n\n");
     printf("-----------------------------------------
          ----------------------------------\n");
     printf("Month    Observed       Trend Seasonal
          Cyclical    Three     Three   Irragular\n");
     printf("           Values      Values    Index
          Irrelars  Monthly   Monthly  Variations\n");
     printf("                                          (%)
        Moving    Moving         (%)\n");
     printf("                                        Totals
       Averages     CtRt/Ct\n");
     printf("    X          Yt          Tt           CtRt
         (%)  Ct          Rt\n");
     printf("    I          II         III       IV
          V       VI       VII        VIII\n");
     printf("-----------------------------------------
          ----------------------------------\n");
     for(i=1;i<=12;i++)
          {
          printf("%5d%12.2lf%12.2lf%9.2lf %9.2lf%9.2lf
               %9.2lf  %9.4lf\n", i, avgamt[i],trend[i],
               adjsidx[i],cycirr[i],mtot3[i],
               mavg3[i],irrvar[i]);
          fprintf(fp2,"%d,%lf,%lf,%lf,%lf,%lf,%lf\n",
               i,mavgamt[i],trend[i],adjsidx[i],
               cycirr[i],mavg3[i],irrvar[i]);
          }
     printf("-----------------------------------------
          ----------------------------------\n");
     printf("Press any key to continue...\n");
     getch();
     clrscr();
}
```

7. lsq2nd.cpp

```
/*
Monthly, Weekly number of hits for time series
1st week- Date 1 to 7
2nd week= Date 8 to 14
3rd week- Date 16 to 21
Last week- Date 22 to end of the month
Program to calculate monthly trend using least square
method*/

#include <stdio.h>
# include <conio.h>
main()
{
     FILE *fp1,*fp2,*fp3;
     struct atm
     {
          int sr;
          int yy;
          int mm;
          int dd;
          long int dt;
          double tm;
          double amt;
     };
     struct atm day;
     int tmpmm;
     int i,j,k,hit2[13][5]={{0},{0}},mtothit[13]={0},
          sumhit=0;
     float mavghit[13]={0},xdev[13],xdev2[13],xdevy[13],
          xdev2y[13], xdev3[13],xdev4[13], trend[13];
     float sumy=0,sumxdevy=0,sumx=0,sumxdev=0,sumxdev2=0,
          sumxdev2y=0,   sumxdev3=0,sumxdev4=0,xbar,a,b,
          weekincr;
     clrscr();
     fp1=fopen("ymdtamt.csv","r");
     fp2=fopen("lsq2nd.txt","w");
     fp3=fopen("srtest.txt","w");
     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     i=1;
     while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
          &day.yy,&day.mm,&day.dd,&day.dt,&day.tm,
          &day.amt)!=EOF)
          {
          tmpmm=day.mm;
          if(day.dd>=1 && day.dd<=7)
               j=1;
```

```
        if(day.dd>=8 && day.dd<=14)
             j=2;
        if(day.dd>=15 && day.dd<=21)
             j=3;
        if(day.dd>=22 && day.dd<=31)
             j=4;
        i=tmpmm;
        hit2[i][j]++;
        fprintf(fp3,"%d\n",day.sr);
        while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
             &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
             &day.tm,&day.amt)!=EOF && mpmm==day.mm)
             {
             if(day.dd>=1 && day.dd<=7)
                  j=1;
             if(day.dd>=8 && day.dd<=14)
                  j=2;
             if(day.dd>=15 && day.dd<=21)
                  j=3;
             if(day.dd>=22 && day.dd<=31)
                  j=4;
             i=tmpmm;
             hit2[i][j]++;
             fprintf(fp3,"%d\n",day.sr);
             }
        if(day.dd>=1 && day.dd<=7)
             j=1;
        if(day.dd>=8 && day.dd<=14)
             j=2;
        if(day.dd>=15 && day.dd<=21)
             j=3;
        if(day.dd>=22 && day.dd<=31)
             j=4;
        i=day.mm;
        hit2[i][j]++;
        fprintf(fp3,"%d\n",day.sr);
        }
   hit2[i][j]--;
   fclose(fp1);
   fclose(fp3);
   for(i=1;i<=12;i++)
        {
        for(j=1;j<=4;j++)
             {
             mtothit[i]=mtothit[i]+hit2[i][j];
             }
        mavghit[i]=mtothit[i]/4.0;
        sumhit=sumhit+mtothit[i];
        sumx=sumx+i;
        }
   xbar=sumx/12.0;
```

```
     for(i=1;i<=12;i++)
          {
          xdev[i]=i-xbar;
          xdev2[i]=xdev[i]*xdev[i];
          xdev3[i]=xdev2[i]*xdev[i];
          xdev4[i]=xdev3[i]*xdev[i];
          xdevy[i]=xdev[i]*mavghit[i];
          xdev2y[i]=xdevy[i]*xdev[i];
          sumxdev=sumxdev+xdev[i];
          sumy=sumy+mavghit[i];
          sumxdev2=sumxdev2+xdev2[i];
          sumxdevy=sumxdevy+xdevy[i];
          sumxdev2y=sumxdev2y+xdev2y[i];
          sumxdev3=sumxdev3=xdev3[i];
          sumxdev4=sumxdev4+xdev4[i];
          }
     printf("\n\n  Table Showing feeting of second degree
          curve\n\n");
     printf(" ------------------------------------------
          ----------------------------------\n");
     printf("                WEEK                    \n");
     printf(" Month  -----------------   Avg        \n");
     printf("   X  Xdev   I  II III Last Tot      Y
          Xdev^2  Xdev^3  Xdev^4   XdevY  Xdev^2y\n");
     printf(" ------------------------------------------
          ----------------------------------\n");
     for(i=1;i<=12;i++)
          {
          printf("%4d%6.2f",i,xdev[i]);
          for(j=1;j<=4;j++)
               {
               printf("%4d",hit2[i][j]);
               }
          printf(" %4d %7.2f%7.2f%8.2f%8.2f%8.2f
               %9.2f\n",mtothit[i], mavghit[i], xdev2[i],
               xdev3[i],xdev4[i],xdevy[i],xdev2y[i]);
          fprintf(fp2,"%d%8.2f%8.2f\n",i,xdev[i],
               xdev2[i]);
          }
     printf(" ------------------------------------------
          ----------------------------------\n");
     printf("    %6.2f                %5d %7.2f %6.2f
       %7.2f%8.2f%8.2f %9.2f  \n", sumxdev,sumhit,sumy,
       sumxdev2,sumxdev3,sumxdev4, sumxdevy, sumxdev2y);
     printf(" ------------------------------------------
          ----------------------------------\n");
     printf(" Press any key to continue....\n");
     fclose(fp1);
     getch();
     clrscr();
}
```

8. lsq2sol.cpp

```
#include <stdio.h>
#include <conio.h>
main()
{
     float a[3][4]={     {12, 0, 143, 4456.00},
                         {0,143,166.38,2521.00},
                         {143,166.38,3038.75,51571.50}
                    };
     float t;
     int i, j, k;
     clrscr();
     for(i=0;i<=2;i++)
          {
          t=a[i][i];
          for(j=0;j<=3;j++)
               {
               a[i][j]=a[i][j]/t;
               }
          for(k=0;k<=2;k++)
               {
               if(i==k)
                    continue;
               t=a[k][i];
               for(j=0;j<=3;j++)
                    {
                    a[k][j]=a[k][j]-a[i][j]*t;
                    }
               }
          }
          printf("\n");
          for(i=0;i<=2;i++)
               {
               for(j=0;j<=3;j++)
                    printf("%12.6f  ",a[i][j]);
               printf("\n\n");
               }
     getch();
}
```

9. lsq2ndam.cpp

```
/*
Monthly, Weekly amount withdrawn for time series
1st week- Date 1 to 7
2nd week= Date 8 to 14
3rd week- Date 16 to 21
Last week- Date 22 to end of the month
```

```
Program to calculate monthly trend using least square
method*/

#include <stdio.h>
# include <conio.h>
main()
{
     FILE *fp1,*fp2,*fp3;
     struct atm
     {
          int sr;
          int yy;
          int mm;
          int dd;
          long int dt;
          double tm;
          double amt;
     };
     struct atm day;
     int tmpmm;
     int i,j,k;
     long int amt2[13][5]={{0},{0}},mtotamt[13]={0},
          sumamt=0;
     double mavgamt[13]={0},xdev[13],xdev2[13],xdevy[13],
          xdev2y[13], xdev3[13],xdev4[13],trend[13];
     double sumy=0,sumxdevy=0,sumx=0,sumxdev=0,
          sumxdev2=0,sumxdev2y=0,sumxdev3=0,sumxdev4=0,
          xbar,a,b,weekincr;
     clrscr();
     fp1=fopen("ymdtamt.csv","r");
     fp2=fopen("lsq2ndam.txt","w");
     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     i=1;
     while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
          &day.yy,&day.mm,&day.dd,&day.dt,&day.tm,
          &day.amt)!=EOF)
          {
          tmpmm=day.mm;
          while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
               &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
               &day.tm,&day.amt)!=EOF&&tmpmm==day.mm)
               {
               if(day.dd>=1 && day.dd<=7)
                    j=1;
               if(day.dd>=8 && day.dd<=14)
                    j=2;
               if(day.dd>=15 && day.dd<=21)
                    j=3;
```

```
          if(day.dd>=22 && day.dd<=31)
               j=4;
          i=tmpmm;
          amt2[i][j]=amt2[i][j]+day.amt;
          }
     }
fclose(fp1);
for(i=1;i<=12;i++)
     {
     for(j=1;j<=4;j++)
          {
          mtotamt[i]=mtotamt[i]+amt2[i][j];
          }
     mavgamt[i]=mtotamt[i]/4.0;
     sumamt=sumamt+mtotamt[i];
     sumx=sumx+i;
     }
xbar=sumx/12.0;
for(i=1;i<=12;i++)
     {
     xdev[i]=i-xbar;
     xdev2[i]=xdev[i]*xdev[i];
     xdev3[i]=xdev2[i]*xdev[i];
     xdev4[i]=xdev3[i]*xdev[i];
     xdevy[i]=xdev[i]*mavgamt[i];
     xdev2y[i]=xdevy[i]*xdev[i];
     sumxdev=sumxdev+xdev[i];
     sumy=sumy+mavgamt[i];
     sumxdev2=sumxdev2+xdev2[i];
     sumxdevy=sumxdevy+xdevy[i];
     sumxdev2y=sumxdev2y+xdev2y[i];
     sumxdev3=sumxdev3=xdev3[i];
     sumxdev4=sumxdev4+xdev4[i];
     }
printf("\n\n  Table showing feeting of second degree
     curve\n\n");
printf(" ------------------------------------------
     --------------------------\n");
printf("                                   WEEK
          \n");
printf(" Month         ------------------------------
     -----                    Avg \n");
printf("     X   Xdev         I       II      III
   Last       Tot           Y \n");
printf(" ------------------------------------------
     --------------------------\n");
for(i=1;i<=12;i++)
     {
     printf("  %4d %6.2lf",i,xdev[i]);
     for(j=1;j<=4;j++)
          {
```

```
                printf("%9ld",amt2[i][j]);
                }
           printf(" %9ld%12.2lf\n",mtotamt[i],mavgamt[i]);
           }
     printf(" ------------------------------------------
           -------------------------\n");
     printf("       %6.2lf                    %9ld\n",
           sumxdev,  sumamt);
     printf(" ------------------------------------------
           -------------------------\n");
     printf("\n\n  Table showing feeting of second degree
           curve\n\n");
     printf(" ------------------------------------------
           -------------------------------\n");
     printf(" Month               Avg\n");
     printf("     X  Xdev           Y  Xdev^2  Xdev^3
       Xdev^4         XdevY       Xdev^2y\n");
     printf(" ------------------------------------------
           -------------------------------\n");
     for(i=1;i<=12;i++)
           {
           printf("  %4d%6.2lf",i,xdev[i]);
           printf("%12.2lf%8.2lf%8.2lf%8.2lf%14.2lf
           %14.2lf\n",mavgamt[i],xdev2[i],xdev3[i],
           xdev4[i],xdevy[i],xdev2y[i]);
           fprintf(fp2,"%d%8.2lf%8.2lf\n",i,xdev[i],
           xdev2[i]);
           }
     printf(" ------------------------------------------
           -------------------------------\n");
     printf("      %6.2lf%12.2lf%8.2lf%8.2lf%8.2lf%14.2lf
           %14.2lf\n", sumxdev,sumy, sumxdev2,sumxdev3,
           sumxdev4,sumxdevy,sumxdev2y);
     printf(" ------------------------------------------
           -------------------------------\n");
     printf(" Press any key to continue....\n");
     fclose(fp1);
     getch();
     clrscr();
}
```

10. lsq2sola.cpp

```
#include <stdio.h>
#include <conio.h>
main()
{
     float a[3][4]=
     {    {12, 0, 143, 15922262.5},
          {0,143,166.38,9698503.75},
          {143,166.38,3038.75,187656900.62}
```

```
      };
      float t;
      int i, j, k;
      clrscr();
      for(i=0;i<=2;i++)
            {
            t=a[i][i];
            for(j=0;j<=3;j++)
                  {
                  a[i][j]=a[i][j]/t;
                  }
            for(k=0;k<=2;k++)
                  {
                  if(i==k)
                        continue;
                  t=a[k][i];
                  for(j=0;j<=3;j++)
                        {
                        a[k][j]=a[k][j]-a[i][j]*t;
                        }
                  }
            }
            printf("\n\n");
            for(i=0;i<=2;i++)
                  {
                  for(j=0;j<=3;j++)
                        printf("%15.6f  ",a[i][j]);
                  printf("\n\n");
                  }
      getch();
}
```

11. emimmly.cpp

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
main()
{
      float p=1000000,r,ir,emi;
      int n,nday;
      clrscr();
      printf("\n\n          Home loan EMIs for
            Rs.1000000/- \n");
      printf("             (reducig monthly basis)\n");
      printf("-----------------------------------------
            ---\n");
      printf("   Rate     5 Yrs    10 Yrs    15 Yrs    20
             Yrs\n");
      printf("-----------------------------------------
            ---\n");
```

```
     for(ir=7;ir<=15;ir=ir+0.25)
          {
          printf("  %5.2f",ir);
          for(n=5;n<=20;n=n+5)
               {
               r=ir/1200.00;
               nday=n*12;
               emi=(p*r*pow(1+r,nday)/(pow(1+r,nday)-1));
               printf("%10.2f",emi);
               }
          printf("\n");
          }
     getch();
}
```

12. emidaily.cpp

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
main()
{
     double p=1000000,r,ir,emi,n,nday;
     clrscr();
     printf("\n\n          Home loan EMIs for
          Rs.1000000/- \n");
     printf("              (reducig daily basis)\n");
     printf("-----------------------------------------
          ---\n");
     printf("   Rate     5 Yrs    10 Yrs    15 Yrs    20
          Yrs\n");
     printf("-----------------------------------------
          ---\n");
     for(ir=7;ir<=15;ir=ir+0.25)
          {
          printf("  %5.2lf",ir);
          for(n=5;n<=20;n=n+5)
               {
               r=ir/36500;
               nday=n*365.00;
               emi=(p*r*pow(1+r,nday)/
                    (pow(1+r,nday)-1))*(365/12.0);
               printf("%10.2lf",emi);
               }
          printf("\n");
          }
     getch();
}
```

13. emidiff.cpp

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
main()
{
     float p=1000000,r,ir,emimm,emidd,diff;
     int n,nday;
     FILE *fp;
     clrscr();
     fp=fopen("emidiff.txt","w");
     printf("\n\n                              Home loan of
      Rs.1000000/- \n");
     printf("  Statement showing EMIs on reducig monthly
          basis, daily basis  and difference\n");
     printf(" ----------------------------------------
          ----------------------------------\n");
     printf("               5 Yrs            10 Yrs
           15 Yrs           20 Yrs\n");
     printf("        ---------------------------------
          ----------------------------------\n");
     printf(" Rate  mnthly daily diff mnthly daily diff
          mnthly daily diff   mnthly daily diff\n");
     printf(" ----------------------------------------
          ----------------------------------\n");
     for(ir=7;ir<=15;ir=ir+0.25)
          {
          printf(" %5.2f",ir);
          fprintf(fp,"%5.2f,",ir);
          for(n=5;n<=20;n=n+5)
               {
               r=ir/1200;
               nday=n*12;
               emimm=(p*r*pow(1+r,nday)/
                    (pow(1+r,nday)-1));
               r=ir/36500;
               nday=n*365;
               emidd=(p*r*pow(1+r,nday)/
                    (pow(1+r,nday)-1))*(365/12.0);
               diff=(emimm-emidd)*n*12;
               printf("%7.f%6.f%5.f",emimm,emidd,diff);
               fprintf(fp,"%7.f,%7.f,%7.f,",
                    emimm,emidd,diff);
               }
          printf("\n");
          fprintf(fp,"\n");
          }
     fclose(fp);
     getch();
}
```

14. emicheck.cpp

```
# include <math.h>
# include <stdio.h>
# include <conio.h>
main()
{
     int i;
     double p,r,n,n1,e,eamt,openbal,adjp,adjint,closbal;
     clrscr();
     p=1000000;
     r=11;
     n=5;
     n1=n*12;
     e=21742.00;
     eamt=e*pow((1.00+r/36500.00),(365.00/24.00));
     printf("\n\n           Loan amount:%10.2lf\n",p);
     printf("      Rate of Interest:%10.2lf\n",r);
     printf("  EMI on Monthly basis:%10.2lf\n",e);
     printf("  EMI+15 days interest:%10.2lf\n",eamt);
     openbal=p;
     printf(" ------------------------------------------
          -----------\n");
     printf(" Month      Opening    Adjusted    Adjusted
          Closing\n");
     printf("            Balance   Principal    Interest
          Balance\n");
     printf(" ------------------------------------------
          -----------\n");
     for(i=1;i<=n1;i++)
          {
          adjint=openbal*pow((1+r/36500.00),(365.00/12))-
               openbal;
          adjp=eamt-adjint;
          closbal=openbal-adjp;
          printf(" %5d %12.2lf%12.2lf%12.2lf%12.2lf\n",
               i,openbal,adjp,adjint,closbal);
          openbal=closbal;
          if(i%30==0)
               {
               getch();
               clrscr();
               printf(" ---------------------------------
                    --------------------\n");
               printf(" Month      Opening    Adjusted
                  Adjusted                  Closing\n");
               printf("            Balance   Principal
                  Interest     Balance\n");
               printf(" ---------------------------------
```

```
                    ---------------------\n");
               }
          }
     getch();
}
```

15. emichkal.cpp

```
# include <math.h>
# include <stdio.h>
# include <conio.h>
main()
{
     int i,j,k;
     double p,r,n,n1,e,eamt,openbal,adjp,adjint,
          closbal,perc;
     double emi[2][4]=
          {    21742.00,13775.00,11366.00,10322.00,
               21670.00,13742.00,11346.00,10309.00};
     clrscr();
     r=11;
     printf("\n ----------------------------------------
          ------------------------------\n");
     printf("       5          10          15
             20\n");
     printf("     years       years       years
            years\n");
     printf(" ----------------------------------------
          ------------------------------\n");
     for(i=0;i<=1;i++)          // 0:Monthly, 1:Daily
          {
          printf("\n          EMI            ");
          for(n=5;n<=20;n=n+5)
               {
               p=1000000;
               n1=n*12;
               j=n/5-1;
               e=emi[i][j];
               printf("%12.2lf",e);
               }
          printf("\n");
          if(i==0)
               printf(" Monthly ");
          else
               printf(" Daily   ");

          printf("\n          Closing Balance");
          for(n=5;n<=20;n=n+5)
               {
               p=1000000;
               n1=n*12;
```

```
                j=n/5-1;
                e=emi[i][j];
                eamt=e*pow((1.00+r/36500.00),
                          (365.00/24.00));
                openbal=p;
                for(k=1;k<=n1;k++)
                     {
                     adjint=openbal*pow((1+r/36500.00),
                          (365.00/12))- openbal;
                     adjp=eamt-adjint;
                     closbal=openbal-adjp;
                     openbal=closbal;
                     }
                printf("%12.2lf",closbal);
                }
           printf("\n\n");
           }
     printf(" ------------------------------------------
          ----------------------------\n");
     getch();
}
```

16. fdqtrly.cpp

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
main()
{
     float p=100000,r,ir,amtq,nt;
     int n;
     clrscr();
     printf("\n\n                          Fixed Deposite of
          Rs.100000/- \n");
     printf("        Statement showing Amount returned on
          Quarterly compounded basis\n");
     printf(" ------------------------------------------
          --------------------------------\n");
     printf("                         PERIOD IN MONTHS\n");
     printf("      ------------------------------------
          --------------------------------\n");
     printf("  Rate        12        15        18
            21        24       27        30\n");
     printf(" ------------------------------------------
          --------------------------------\n");
     for(ir=6;ir<=10;ir=ir+0.25)
          {
          printf(" %5.2f",ir);
          for(n=12;n<=30;n=n+3)
               {
               r=ir/400;
```

```
                nt=4.0*(n/12.0);
                amtq=p*pow(1+r,nt);
                printf("%10.2f",amtq);
                }
          printf("\n");
          }
     printf(" ------------------------------------------
          ------------------------------\n");
     printf("  Rate        33        36        39
          42        45       48        51\n");
     printf(" ------------------------------------------
          ------------------------------\n");
     for(ir=6;ir<=10;ir=ir+0.25)
          {
          printf(" %5.2f",ir);
          for(n=33;n<=51;n=n+3)
                {
                r=ir/400;
                nt=4.0*(n/12.0);
                amtq=p*pow(1+r,nt);
                printf("%10.2f",amtq);
                }
          printf("\n");
          }
     getch();
}
```

17. fddaily.cpp

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
main()
{
     float p=100000,r,ir,amtd,nt;
     int n;
     clrscr();
     printf("\n\n     Fixed Deposite of Rs.100000/- \n");
     printf("        Statement showing Amount returned on
          Daily compounded basis\n");
     printf(" ------------------------------------------
          ------------------------------\n");
     printf("                          PERIOD IN MONTHS\n");
     printf("      ------------------------------------
          ------------------------------\n");
     printf("  Rate        12        15        18      21
           24       27        30\n");
     printf(" ------------------------------------------
          ------------------------------\n");
     for(ir=6;ir<=10;ir=ir+0.25)
          {
```

```
          printf(" %5.2f",ir);
          for(n=12;n<=30;n=n+3)
               {
               r=ir/36500;
               nt=365.0*(n/12.0);
               amtd=p*pow(1+r,nt);
               printf("%10.2f",amtd);
               }
          printf("\n");
          }
     printf(" ------------------------------------------
          --------------------------------\n");
     printf("  Rate        33        36        39      42
            45        48        51\n");
     printf(" ------------------------------------------
          --------------------------------\n");
     for(ir=6;ir<=10;ir=ir+0.25)
          {
          printf(" %5.2f",ir);
          for(n=33;n<=51;n=n+3)
               {
               r=ir/36500;
               nt=365.0*(n/12.0);
               amtd=p*pow(1+r,nt);
               printf("%10.2f",amtd);
               }
          printf("\n");
          }
     getch();
}
```

18. fddiff.cpp

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
main()
{
     float p=100000,r,ir,amtd,amtq,diff,nt;
     int n;
     clrscr();
     printf("\n\n     Fixed Deposite of Rs.100000/- \n");
     printf("         Statement showing difference of
          maturity amount returned on\n");
     printf("                      Daily and Quarterly
          compounded basis\n");
     printf(" ------------------------------------------
          --------------------------------\n");
     printf("                        PERIOD IN MONTHS\n");
     printf("     ----------------------------------
          --------------------------------\n");
```

```
     printf("  Rate        12        15        18      21
            24       27        30\n");
     printf(" ------------------------------------------
          ------------------------------\n");
     for(ir=6;ir<=10;ir=ir+0.25)
          {
          printf(" %5.2f",ir);
          for(n=12;n<=30;n=n+3)
               {
               r=ir/36500;
               nt=365.0*(n/12.0);
               amtd=p*pow(1+r,nt);
               r=ir/400;
               nt=4.0*(n/12.0);
               amtq=p*pow(1+r,nt);
               diff=amtd-amtq;
               printf("%10.2f",diff);
               }
          printf("\n");
          }
     printf(" ------------------------------------------
          ------------------------------\n");
     printf("  Rate        33        36        39      42
            45       48        51\n");
     printf(" ------------------------------------------
          ------------------------------\n");
     for(ir=6;ir<=10;ir=ir+0.25)
          {
          printf(" %5.2f",ir);
          for(n=33;n<=51;n=n+3)
               {
               r=ir/36500;
               nt=365.0*(n/12.0);
               amtd=p*pow(1+r,nt);
               r=ir/400;
               nt=4.0*(n/12.0);
               amtq=p*pow(1+r,nt);
               diff=amtd-amtq;
               printf("%10.2f",diff);
               }
          printf("\n");
          }
     getch();
}
```

19. recurqfl.cpp

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
```

```
{
     float p,roi,amt;
     int mnths,i,n=5,yr;
     void headings(FILE *,int);
     FILE *fp1;
     fp1=fopen("recurqfl.txt","w");
     p=100.00;
     for(yr=1;yr<=n;yr++)
          {
          clrscr();
          headings(fp1,yr);
          for(roi=6;roi<=14;roi=roi+0.25)
               {
               fprintf(fp1,"%5.2f,",roi);
               for(mnths=(yr-1)*12+3;mnths<=yr*12;
                                        mnths=mnths+3)
                    {
                    amt=0;
                    for(i=mnths;i>=1;i--)
                         {
                         amt=amt+p*pow((1+roi/400),
                                        (4*i/12.00));
                         }
                    fprintf(fp1,"%12.3f,",amt);
                    }
               fprintf(fp1,"\n");
               }
          }
}
void headings(FILE *fp1,int yr)
{
     int mnths;
     fprintf(fp1," Rate,");
     for(mnths=(yr-1)*12+3;mnths<=yr*12;mnths=mnths+3)
          fprintf(fp1,"%12d,",mnths);
     fprintf(fp1,"\n");
}
```

20. recurdfl.cpp

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
     float p,roi,amt;
     int mnths,i,n=5,yr;
     void headings(FILE *,int);
     FILE *fp1;
     fp1=fopen("recurdfl.txt","w");
     p=100.00;
```

```
     for(yr=1;yr<=n;yr++)
          {
          clrscr();
          headings(fp1,yr);
          for(roi=6;roi<=14;roi=roi+0.25)
               {
               fprintf(fp1,"%5.2f,",roi);
               for(mnths=(yr-1)*12+3;mnths<=yr*12;
                                        mnths=mnths+3)
                    {
                    amt=0;
                    for(i=mnths;i>=1;i--)
                         {
                         amt=amt+p*pow((1+roi/36500),
                                        (365*i/12.00));
                         }
                    fprintf(fp1,"%12.3f,",amt);
                    }
               fprintf(fp1,"\n");
               }
          }
}
void headings(FILE *fp1,int yr)
{
     int mnths;
     fprintf(fp1," Rate,");
     for(mnths=(yr-1)*12+3;mnths<=yr*12;mnths=mnths+3)
          fprintf(fp1,"%12d,",mnths);
     fprintf(fp1,"\n");
}
```

21. recudiff.cpp

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
     float p,roi,amtq,amtd,diff;
     int mnths,i,n=5,yr;
     void headings(FILE *,int);
     FILE *fp1;
     fp1=fopen("recudiff.txt","w");
     p=100.00;
     for(yr=1;yr<=n;yr++)
          {
          clrscr();
          headings(fp1,yr);
          for(roi=6;roi<=14;roi=roi+0.25)
               {
               fprintf(fp1,"%5.2f,",roi);
```

```
                for(mnths=(yr-1)*12+3;mnths<=yr*12;
                                        mnths=mnths+3)
                     {
                     amtq=amtd=0;
                     for(i=mnths;i>=1;i--)
                          {
                          amtd=amtd+p*pow((1+roi/36500),
                                        (365*i/12.00));
                          amtq=amtq+p*pow((1+roi/400),
                                        (4*i/12.00));
                          }
                     diff=amtd-amtq;
                     fprintf(fp1,"%12.3f,",diff);
                     }
                fprintf(fp1,"\n");
                }
           }
}
void headings(FILE *fp1,int yr)
{
     int mnths;
     fprintf(fp1," Rate,");
     for(mnths=(yr-1)*12+3;mnths<=yr*12;mnths=mnths+3)
          fprintf(fp1,"%12d,",mnths);
     fprintf(fp1,"\n");
}
```

22. grfile.cpp

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
     FILE *fp1,*fp2;
     long double ts,tt,sbar,tbar,tmp;
     long double t0,b0,tb01,tb02;
     long double as[366],at[366],ab01[366],ab02[366];
     long double ay4[366],ty4;
     /*   double tmp;*/
     int i,x,ax[366],tx,n=365,tmpi;

     clrscr();

     fp1=fopen("snrdhita.csv","r");
     fp2=fopen("graphicl.txt","w");

     if(fp1==NULL)
     {
          puts("cannot open the file");
```

```
}
tmpi=0;
tx=0;
ts=0.0;
tt=0.0;
while(fscanf(fp1,"%d,%d",&i,&x)!=EOF)
     {
     if(tmpi==i)
          break;
     ax[i]=x;
     as[i]=log(log((n+1)/(float)(n-i+1)));
     at[i]=log(x);
     tx=tx+x;
     ts=ts+as[i];
     tt=tt+at[i];
     tmpi=i;
     }
sbar=ts/n;
tbar=tt/n;
fclose(fp1);
tb01=tb02=0.0;
for(i=1;i<=n;i++)
     {
     ab01[i]=(as[i]-sbar)*(at[i]-tbar);
     tb01=tb01+ab01[i];
     ab02[i]=(at[i]-tbar)*(at[i]-tbar);
     tb02=tb02+ab02[i];
     }
b0=tb01/tb02;

fprintf(fp2,"----------------------------------------
     ----------------------------------------------
     ---------\n");
fprintf(fp2,"    i    x           si           ti
    (si-sbar)(ti-tbar)     (ti-tbar)^2    xi^b0\n");
fprintf(fp2,"----------------------------------------
     ----------------------------------------------
     ---------\n");
ty4=0.0;
for(i=1;i<=n;i++)
     {
     tmp=ax[i];
     ay4[i]=pow(tmp,b0);
     ty4=ty4+ay4[i];
     fprintf(fp2,"%5d%5d%13.6Lf%13.6Lf
        %13.6Lf   %13.6Lf          %20.6Lf\n",
        i,ax[i],as[i],at[i],ab01[i],ab02[i],ay4[i]);
     }
fprintf(fp2,"----------------------------------------
     ----------------------------------------------
     ---------\n");
```

```
     fprintf(fp2,"TOT  %5d%13.6Lf%13.6Lf          %13.6Lf
          %13.6Lf  %20.6Lf\n",tx,ts,tt,tb01,tb02,ty4);
     fprintf(fp2,"----------------------------------------
          ----------------------------------------------
          ---------\n");
     fprintf(fp2,"          %13.6Lf%13.6Lf\n",sbar,tbar);
     fprintf(fp2,"----------------------------------------
          ----------------------------------------------
          ---------\n");
     t0=ty4/(long double)n;
     fprintf(fp2,"\nBeta=%10.7Lf  Theta=%17.6Lf\n"
                                          ,b0,t0);
     fclose(fp2);
}
```

23. nrtable1.cpp

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
     FILE *fp1,*fp2;
     long double ts,tt,sbar,tbar,tmp;
     long double z1,z2,t0,b0,b1,diff;
     long double tb01,tb02;
     long double as[366],at[366],ab01[366],ab02[366];
     long double ay1[366],ay2[366],ay3[366],ay4[366],
                                        ty1,ty2,ty3,ty4;
     /*   double tmp;*/
     int i,x,ax[366],tx,n=365,tmpi,iteration=1;

     clrscr();

     fp1=fopen("snrdhita.csv","r");
     fp2=fopen("nriterja.txt","w");

     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     tmpi=0;
     tx=0;
     ts=0.0;
     tt=0.0;
     while(fscanf(fp1,"%d,%d",&i,&x)!=EOF)
          {
          if(tmpi==i)
               break;
          ax[i]=x;
          as[i]=log(log((n+1)/(float)(n-i+1)));
```

```
     at[i]=log(x);
     tx=tx+x;
     ts=ts+as[i];
     tt=tt+at[i];
     tmpi=i;
     }
sbar=ts/n;
tbar=tt/n;
fclose(fp1);
tb01=tb02=0.0;
for(i=1;i<=n;i++)
     {
     ab01[i]=(as[i]-sbar)*(at[i]-tbar);
     tb01=tb01+ab01[i];
     ab02[i]=(at[i]-tbar)*(at[i]-tbar);
     tb02=tb02+ab02[i];
     }
b0=tb01/tb02;

diff=1.00;
iteration=1;

while(diff>=0.0001)
     {
     fprintf(fp2,"\n\nIteration: %d\n",iteration);
     fprintf(fp2,"-----------------------------------
          -----------------------------------------
          -----------------------------------------
          ------------------------\n");
     fprintf(fp2,"    i    x      si      ti     b01
         b02     y1      y2      y3       y4\n");
     fprintf(fp2,"-----------------------------------
          -----------------------------------------
          -----------------------------------------
          ------------------------\n");
     ty1=0.0;
     ty2=0.0;
     ty3=0.0;
     ty4=0.0;
     for(i=1;i<=n;i++)
          {
          tmp=ax[i];
          ay4[i]=pow(tmp,b0);
          ay1[i]=log(tmp);
          ay2[i]=ay4[i]*ay1[i];
          ay3[i]=ay2[i]*ay1[i];
          ty4=ty4+ay4[i];
          ty1=ty1+ay1[i];
          ty2=ty2+ay2[i];
          ty3=ty3+ay3[i];
          fprintf(fp2,"%5d%5d%13.6Lf%13.6Lf%13.6Lf
```

```
                    %13.6Lf%20.6Lf%20.6Lf%20.6Lf
                    %20.6Lf\n",i,ax[i],as[i],at[i],
                    ab01[i],ab02[i], ay1[i],
                    ay2[i],ay3[i],ay4[i]);
               }
          fprintf(fp2,"----------------------------------
               ----------------------------------------
               ----------------------------------------
               ------------------------\n");
          fprintf(fp2,"TOT  %5d%13.6Lf%13.6Lf%13.6Lf
               %13.6Lf%20.6Lf%20.6Lf%20.6Lf%20.6Lf\n",
               tx,ts,tt,tb01,tb02,ty1,ty2,ty3,ty4);
          fprintf(fp2,"----------------------------------
               ----------------------------------------
               ----------------------------------------
               ------------------------\n");
          fprintf(fp2,"          %13.6Lf%13.6Lf\n",
                                        sbar,tbar);
          fprintf(fp2,"----------------------------------
               ----------------------------------------
               ----------------------------------------
               ------------------------\n");
          t0=ty4/(long double)n;
          z1=((long double)n/b0)+ty1-(ty2/t0);
          z2=(-n/pow(b0,2))-(n*ty3/ty4);
          b1=b0-z1/z2;
          diff=fabs(b1-b0);
          fprintf(fp2,"\nb0=%10.7Lf  b1=%10.7Lf
                difference=%10.7Lf  theta= %17.6Lf\n",
                                    b0,b1,diff,t0);
          b0=b1;
          iteration++;
          }
     fclose(fp2);
}
```

24. gramt.cpp

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
     FILE *fp1,*fp2;
     long double ts,tt,sbar,tbar,tmp;
     long double t0,b0,tb01,tb02;
     long double as[366],at[366],ab01[366],ab02[366];
     long double ay4[366],ty4,x,ax[366],tx;
     /*   double tmp;*/
     int i,n=365,tmpi;

     clrscr();
```

```
fp1=fopen("snramta.csv","r");
fp2=fopen("graphamt.txt","w");

if(fp1==NULL)
{
     puts("cannot open the file");
}
tmpi=0;
tx=0;
ts=0.0;
tt=0.0;
while(fscanf(fp1,"%d,%Lf",&i,&x)!=EOF)
     {
     if(tmpi==i)
          break;
     ax[i]=x;
     as[i]=log(log((n+1)/(float)(n-i+1)));
     at[i]=log(x);
     tx=tx+x;
     ts=ts+as[i];
     tt=tt+at[i];
     tmpi=i;
     }
sbar=ts/n;
tbar=tt/n;
fclose(fp1);
tb01=tb02=0.0;
for(i=1;i<=n;i++)
     {
     ab01[i]=(as[i]-sbar)*(at[i]-tbar);
     tb01=tb01+ab01[i];
     ab02[i]=(at[i]-tbar)*(at[i]-tbar);
     tb02=tb02+ab02[i];
     }
b0=tb01/tb02;

fprintf(fp2,"----------------------------------------
          ----------------------------------------
          -------------------\n");
fprintf(fp2,"    i        x           si           ti
   (si-sbar)(ti-tbar)   (ti-tbar)^2       xi^b0\n");
fprintf(fp2,"----------------------------------------
     ----------------------------------------------
     -------------\n");
ty4=0.0;
for(i=1;i<=n;i++)
     {
     tmp=ax[i];
     ay4[i]=pow(tmp,b0);
     ty4=ty4+ay4[i];
```

```
          fprintf(fp2,"%5d%10.0Lf%13.6Lf%13.6Lf   %13.6Lf
             %13.6Lf  %25.6Lf\n", i,ax[i],as[i],at[i],
               ab01[i],ab02[i],ay4[i]);
          }
     fprintf(fp2,"-----------------------------------------
          ----------------------------------------------
          --------------\n");
     fprintf(fp2,"TOT %11.0Lf%13.6Lf%13.6Lf   %13.6Lf
       %13.6Lf  %25.6Lf\n", tx,ts,tt,tb01,tb02,ty4);
     fprintf(fp2,"-----------------------------------------
          ----------------------------------------------
          --------------\n");
     fprintf(fp2,"          %13.6Lf%13.6Lf\n",sbar,tbar);
     fprintf(fp2,"-----------------------------------------
          ----------------------------------------------
          --------------\n");
     t0=ty4/(long double)n;
     fprintf(fp2,"\nBeta=%10.7Lf  Theta=%17.6Lf\n",
                                                  b0,t0);
     printf("\nBeta=%10.7Lf  Theta=%17.6Lf\n",b0,t0);
     fclose(fp2);
     getch();
}
```

25. nramt.cpp

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
     FILE *fp1,*fp2;
     long double ts,tt,sbar,tbar,tmp;
     long double z1,z2,t0,b0,b1,diff;
     long double tb01,tb02,x,ax[366],tx;
     long double as[366],at[366],ab01[366],ab02[366];
     long double ay1[366],ay2[366],ay3[366],ay4[366],
                                        ty1,ty2,ty3,ty4;
     /*   double tmp;*/
     int i,n=365,tmpi,iteration=1;

     clrscr();

     fp1=fopen("snramta.csv","r");
     fp2=fopen("nramt.txt","w");

     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     tmpi=0;
```

```
tx=0;
ts=0.0;
tt=0.0;
while(fscanf(fp1,"%d,%Lf",&i,&x)!=EOF)
     {
     if(tmpi==i)
          break;
     ax[i]=x;
     as[i]=log(log((n+1)/(float)(n-i+1)));
     at[i]=log(x);
     tx=tx+x;
     ts=ts+as[i];
     tt=tt+at[i];
     tmpi=i;
     }
sbar=ts/n;
tbar=tt/n;
fclose(fp1);
tb01=tb02=0.0;
for(i=1;i<=n;i++)
     {
     ab01[i]=(as[i]-sbar)*(at[i]-tbar);
     tb01=tb01+ab01[i];
     ab02[i]=(at[i]-tbar)*(at[i]-tbar);
     tb02=tb02+ab02[i];
     }
b0=tb01/tb02;

diff=1.00;
iteration=1;

while(diff>=0.0001)
     {
     fprintf(fp2,"\n\nIteration: %d\n",iteration);
     fprintf(fp2,"-----------------------------------
          ------------------------------------------
          ------------------------------------------
          ------------------------------------\n");
     fprintf(fp2,"    i       x        si      ti
      b01     b02      y1     y2     y3      y4\n");
     fprintf(fp2,"-----------------------------------
          ------------------------------------------
          ------------------------------------------
          ------------------------------------\n");
     ty1=0.0;
     ty2=0.0;
     ty3=0.0;
     ty4=0.0;
     for(i=1;i<=n;i++)
          {
          tmp=ax[i];
```

```
                ay4[i]=pow(tmp,b0);
                ay1[i]=log(tmp);
                ay2[i]=ay4[i]*ay1[i];
                ay3[i]=ay2[i]*ay1[i];
                ty4=ty4+ay4[i];
                ty1=ty1+ay1[i];
                ty2=ty2+ay2[i];
                ty3=ty3+ay3[i];
                fprintf(fp2,"%5d%10.0Lf%13.6Lf%13.6Lf
                   %13.6Lf%13.6Lf%13.6Lf %27.6Lf%27.6Lf
                   %27.6Lf\n",i,ax[i],as[i],at[i],ab01[i],
                   ab02[i],ay1[i],ay2[i],ay3[i],ay4[i]);
                }
           fprintf(fp2,"----------------------------------
                ------------------------------------------
                ------------------------------------------
                ----------------------------------------\n");
           fprintf(fp2,"TOT  %10.0Lf%13.6Lf%13.6Lf%13.6Lf
                %13.6Lf%13.6Lf %27.6Lf%27.6Lf%27.6Lf\n",
                tx,ts,tt,tb01,tb02,ty1,ty2,ty3,ty4);
           fprintf(fp2,"----------------------------------
                ------------------------------------------
                ------------------------------------------
                ----------------------------------------\n");
           fprintf(fp2,"               %13.6Lf%13.6Lf\n",
                                          sbar,tbar);
           fprintf(fp2,"----------------------------------
                ------------------------------------------
                ------------------------------------------
                ----------------------------------------\n");
           t0=ty4/(long double)n;
           z1=((long double)n/b0)+ty1-(ty2/t0);
           z2=(-n/pow(b0,2))-(n*ty3/ty4);
           b1=b0-z1/z2;
           diff=fabs(b1-b0);
           fprintf(fp2,"\nb0=%10.7Lf  b1=%10.7Lf
                 difference=%10.7Lf  theta=%17.6Lf\n",
                                    b0,b1,diff,t0);
           b0=b1;
           iteration++;
           }
     fclose(fp2);
}
```

26. anovahit.cpp

```
/*
Monthly, Weekly number of hits for time series
1st week- Date 1 to 7
2nd week= Date 8 to 14
3rd week- Date 16 to 21
```

```
Last week- Date 22 to end of the month
Generate two dimensional array of hits, Sum ofsquares
between months,
Sum of squares between weeks, The total sum of square,
Error sum of square
and ANOVA table,
Months as rows and weeks as columns
*/


#include <stdio.h>
# include <conio.h>
main()
{
     FILE *fp1;
     struct atm
     {
          int sr;
          int yy;
          int mm;
          int dd;
          long int dt;
          double tm;
          double amt;
     };
     struct atm day;
     int tmpmm;
     int i,j,k,hit2[13][5]={{0},{0}},dofr,dofc,dofe;
     float rtot[13]={0},ctot[5]={0},rtots[13],ctots[5],
          gtot,rtotss=0, ctotss=0,hitss=0;
     float ssr,ssc,cf,sst,sse,msr,msc,mse,
                    fratior,fratioc;
     clrscr();
     fp1=fopen("ymdtamt.csv","r");
     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     i=1;
     while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
          &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
          &day.tm,&day.amt)!=EOF)
          {
          tmpmm=day.mm;
          if(day.dd>=1 && day.dd<=7)
               j=1;
          if(day.dd>=8 && day.dd<=14)
               j=2;
          if(day.dd>=15 && day.dd<=21)
               j=3;
          if(day.dd>=22 && day.dd<=31)
```

```
                j=4;
          i=tmpmm;
          hit2[i][j]++;
          while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
                &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
                &day.tm,&day.amt)!=EOF && tmpmm==day.mm)
                {
                if(day.dd>=1 && day.dd<=7)
                     j=1;
                if(day.dd>=8 && day.dd<=14)
                     j=2;
                if(day.dd>=15 && day.dd<=21)
                     j=3;
                if(day.dd>=22 && day.dd<=31)
                     j=4;
                i=tmpmm;
                hit2[i][j]++;
                }
          tmpmm=day.mm;
          if(day.dd>=1 && day.dd<=7)
                j=1;
          if(day.dd>=8 && day.dd<=14)
                j=2;
          if(day.dd>=15 && day.dd<=21)
                j=3;
          if(day.dd>=22 && day.dd<=31)
                j=4;
          i=tmpmm;
          hit2[i][j]++;
          }
     hit2[i][j]--;
     fclose(fp1);
/*
TO CALCULATE ROW TOTALS, ITS SQUARES AND SUM OF SQUARES
*/
     for(i=1;i<=12;i++)
          {
          for(j=1;j<=4;j++)
                {
                rtot[i]=rtot[i]+hit2[i][j];
                }
          rtots[i]=rtot[i]*rtot[i];
          rtotss=rtotss+rtots[i];
          }
/*
TO CALCULATE COLUMN TOTALS, ITS SQUARES AND SUM OF
SQUARES
*/
     for(j=1;j<=4;j++)
          {
          for(i=1;i<=12;i++)
```

```
                {
                ctot[j]=ctot[j]+hit2[i][j];
                }
          ctots[j]=ctot[j]*ctot[j];
          ctotss=ctotss+ctots[j];
          }
/*
TO CALCULATE WEEKLY HITS SUM OF SQUARES (12X4) AND GRAND
TOTAL
*/
     for(i=1;i<=12;i++)
          {
          for(j=1;j<=4;j++)
                {
                hitss=hitss+(float)hit2[i][j]*hit2[i][j];
                gtot=gtot+hit2[i][j];
                }
          }
/*
TO CALCULATE CF, MONTHSS(SSR), HITSS(SSC), TOTAL SS(SST)
AND ERROR SS (SSE)
*/
     cf=gtot*gtot/48.0;
     ssr=rtotss/4-cf;
     ssc=ctotss/12-cf;
     sst=hitss-cf;
     sse=sst-ssr-ssc;

     printf("\n\n        Table showing monthwise weekly
          hits and sum of squares\n");
     printf(" ------------------------------------------
          ----------------------------------\n");
     printf("                              WEEK         \n");
     printf(" Month        I        II       III     Last
        Row    Sum   Sum of square\n");
     printf(" ------------------------------------------
          ----------------------------------\n");
     for(i=1;i<=12;i++)
          {
          printf("%5d        ",i);
          for(j=1;j<=4;j++)
                {
                printf("%10d",hit2[i][j]);
                }
          printf("  %10.2f    %10.2f\n",
                                rtot[i],rtots[i]);
          }
     printf(" ------------------------------------------
          ----------------------------------\n");
     printf(" Column Sum  %10.0f%10.0f%10.0f%10.0f %11.2f
        %11.2f\n",  ctot[1], ctot[2],ctot[3],ctot[4],
```

```
                                        gtot,rtotss);
     printf(" Sum of Squ. %10.0f%10.0f%10.0f%10.0f
          %11.2f\n",ctots[1], ctots[2],ctots[3],
          ctots[4],ctotss);
     printf(" ----------------------------------------
          ----------------------------------\n");
     printf("        CF = %10.2f      Month SS = %10.2f
       Week SS = %10.2f\n",cf,ssr,ssc);
     printf(" Sum of Hit^2 = %10.2f     Total SS = %10.2f
          Error SS= %10.2f\n",hitss,sst,sse);
//   printf(" Press any key to continue....\n");
//   getch();
//   clrscr();

     dofr=11;
     dofc=3;
     dofe=dofr*dofc;
     msr=ssr/dofr;
     msc=ssc/dofc;
     mse=sse/dofe;
     fratior=msr/mse;
     fratioc=msc/mse;
     printf("\n\n\n The table for Analysis of Variance
          (ANOVA)\n");
     printf(" ----------------------------------------
          ------------------------------\n");
     printf(" Source of                Sum of    Degrees
          of        Mean     F-Ratios\n");
     printf(" variation               squares
           freedom       squares\n");
     printf(" ----------------------------------------
          ------------------------------\n");
     printf(" Between weeks(ssc) %12.2f         %2d
           %12.2f %12.2f\n\n",ssc,dofc,msc,fratioc);
     printf(" Between months(ssr)%12.2f         %2d
            %12.2f %12.2f\n\n", ssr,dofr,msr,fratior);
     printf(" Error(sse)         %12.2f         %2d
          %12.2f\n",sse,dofe,mse);
     printf(" ----------------------------------------
          ------------------------------\n");
     getch();
     clrscr();
}
```

27. anovaamt.cpp

```
/*
Monthly, Weekly amount withdrawn - ANOVA
1st week- Date 1 to 7
2nd week= Date 8 to 14
3rd week- Date 16 to 21
```

```
Last week- Date 22 to end of the month
Generate two dimensional array of hits, Sum ofsquares
between months,
Sum of squares between weeks, The total sum of square,
Error sum of square
and ANOVA table,
Months as rows and weeks as columns
*/


#include <stdio.h>
# include <conio.h>
main()
{
     FILE *fp1;
     struct atm
     {
          int sr;
          int yy;
          int mm;
          int dd;
          long int dt;
          double tm;
          double amt;
     };
     struct atm day;
     int tmpmm;
     int i,j,k,dofr,dofc,dofe;
     double amt2[13][5]={{0},{0}},rtot[13]={0},
          ctot[5]={0},rtots[13], ctots[5],gtot,
          rtotss=0,ctotss=0,amtss=0;
     double ssr,ssc,cf,sst,sse,msr,msc,mse,
          fratior,fratioc;

     clrscr();

     fp1=fopen("ymdtamt.csv","r");

     if(fp1==NULL)
     {
          puts("cannot open the file");
     }
     i=1;
     while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",&day.sr,
          &day.yy,&day.mm,&day.dd,&day.dt,&day.tm,
          &day.amt)!=EOF)
          {
          tmpmm=day.mm;
          while(fscanf(fp1,"%d,%d,%d,%d,%ld,%lf,%lf",
               &day.sr,&day.yy,&day.mm,&day.dd,&day.dt,
               &day.tm,&day.amt)!=EOF && tmpmm==day.mm)
```

```
                {
                if(day.dd>=1 && day.dd<=7)
                     j=1;
                if(day.dd>=8 && day.dd<=14)
                     j=2;
                if(day.dd>=15 && day.dd<=21)
                     j=3;
                if(day.dd>=22 && day.dd<=31)
                     j=4;
                i=tmpmm;
                amt2[i][j]=amt2[i][j] + day.amt;
                }
          }
     fclose(fp1);
/*
TO CALCULATE ROW TOTALS, ITS SQUARES AND SUM OF SQUARES
*/
     for(i=1;i<=12;i++)
          {
          for(j=1;j<=4;j++)
                {
                rtot[i]=rtot[i]+amt2[i][j];
                }
          rtots[i]=rtot[i]*rtot[i];
          rtotss=rtotss+rtots[i];
          }
/*
TO CALCULATE COLUMN TOTALS, ITS SQUARES AND SUM OF
SQUARES
*/
     for(j=1;j<=4;j++)
          {
          for(i=1;i<=12;i++)
                {
                ctot[j]=ctot[j]+amt2[i][j];
                }
          ctots[j]=ctot[j]*ctot[j];
          ctotss=ctotss+ctots[j];
          }
/*
TO CALCULATE WEEKLY HITS SUM OF SQUARES (12X4) AND GRAND
TOTAL
*/
     for(i=1;i<=12;i++)
          {
          for(j=1;j<=4;j++)
                {
                amtss=amtss+amt2[i][j]*amt2[i][j];
                gtot=gtot+amt2[i][j];
                }
          }
```

```
/*
TO CALCULATE CF, MONTHSS(SSR), HITSS(SSC), TOTAL SS(SST)
AND ERROR SS (SSE)
*/
     cf=gtot*gtot/48.0;
     ssr=rtotss/4-cf;
     ssc=ctotss/12-cf;
     sst=amtss-cf;
     sse=sst-ssr-ssc;

     printf("   Table showing monthwise amount withdrawn
          and sum of squares \n");
     printf("-----------------------------------------
          ----------------------------------\n");
     printf("                          WEEK          \n");
     printf("Month      I        II       III      Last
        Row Sum     Row SS\n");
     printf("-----------------------------------------
          ----------------------------------\n");

     for(i=1;i<=12;i++)
          {
          printf("%5d ",i);
          for(j=1;j<=4;j++)
               {
               printf("%10.0lf",amt2[i][j]);
               }
          printf(" %10.0lf  %10.2e\n",rtot[i],rtots[i]);
          }
     printf("-----------------------------------------
          ----------------------------------\n");
     printf("C Sum %10.0lf%10.0lf%10.0lf%10.0lf%11.0lf
          %10.2e\n",ctot[1],ctot[2],ctot[3],ctot[4],
          gtot,rtotss);
     printf("C SS  %10.2e%10.2e%10.2e%10.2e%11.2e\n",
          ctots[1], ctots[2],ctots[3],ctots[4],ctotss);
     printf("-----------------------------------------
          ----------------------------------\n");
     printf("          CF=%10.0lf   Month SS=%10.0lf
           Week SS=%10.0lf\n",
     cf,ssr,ssc);
     printf("Sum of amt^2=%10.0lf   Total SS=%10.0lf
          Error SS=%10.0lf\n", amtss,sst,sse);

     printf("Press any key to continue....\n");
     getch();
     clrscr();

     dofr=11;
     dofc=3;
     dofe=dofr*dofc;
```

```
        msr=ssr/dofr;
        msc=ssc/dofc;
        mse=sse/dofe;
        fratior=msr/mse;
        fratioc=msc/mse;
        printf("\n\n\n      The table for Analysis of
             Variance (ANOVA)\n");
        printf("-----------------------------------------
             ------------------------------\n");
        printf("Source of                    Sum of  Degrees
             of              Mean   F-Ratios\n");
        printf("variation                  squares
            freedom              squares\n");
        printf("-----------------------------------------
             ------------------------------\n");
        printf("Between weeks(ssc)   %12.0lf          %2d
             %16.2lf %10.2lf \n\n", ssc,dofc,msc,fratioc);
        printf("Between months(ssr)  %12.0lf          %2d
              %16.2lf %10.2lf \n\n",ssr,dofr,msr,fratior);
        printf("Error(sse)           %12.0lf          %2d
             %16.2lf        \n",sse,dofe,mse);
        printf("-----------------------------------------
             ------------------------------\n");
        getch();
        clrscr();
}
```

Annexure-II

ATM data

January-2005

Sr.	Date	Time	Amount
1	01/01/05	09:13	8000
2	01/01/05	09:16	1000
3	01/01/05	10:40	100
4	01/01/05	11:23	10000
5	01/01/05	11:37	17000
6	01/01/05	11:50	500
7	01/01/05	11:59	300
8	01/01/05	12:24	6000
9	01/01/05	12:35	1200
10	01/01/05	12:36	10000
11	01/01/05	12:54	1000
12	01/01/05	13:44	8000
13	01/01/05	13:52	2500
14	01/01/05	14:09	5500
15	01/01/05	14:20	2000
16	01/01/05	14:28	4300
17	01/01/05	14:43	200
18	01/01/05	15:28	100
19	01/01/05	16:39	1000
20	01/01/05	16:42	7500
21	01/01/05	16:45	5000
22	01/01/05	16:46	200
23	01/01/05	17:02	3000
24	01/01/05	18:01	100
25	01/01/05	18:15	200
26	01/01/05	18:17	2100
27	01/01/05	18:19	2500
28	01/01/05	18:24	3000
29	01/01/05	18:56	7000
30	01/01/05	19:11	1000
31	01/01/05	19:16	2000
32	01/01/05	19:20	2000
33	01/01/05	19:28	500
34	01/01/05	19:29	100
35	01/01/05	20:02	100
36	01/01/05	20:13	5000
37	01/01/05	20:55	4400
38	01/01/05	21:18	6200
39	01/01/05	21:47	800
40	01/01/05	22:04	2500
41	01/01/05	22:05	3000
42	01/01/05	23:00	100
43	02/01/05	09:08	1000
44	02/01/05	09:41	2000
45	02/01/05	10:32	1000
46	02/01/05	11:06	3000
47	02/01/05	11:07	1000
48	02/01/05	11:27	20000
49	02/01/05	11:29	1000
50	02/01/05	11:34	200
51	02/01/05	11:44	200
52	02/01/05	11:45	200
53	02/01/05	12:23	100
54	02/01/05	12:33	400
55	02/01/05	13:47	500
56	02/01/05	13:50	4900
57	02/01/05	13:52	600
58	02/01/05	13:56	1000
59	02/01/05	14:52	3000
60	02/01/05	16:34	5000
61	02/01/05	17:44	1400
62	02/01/05	17:45	600
63	02/01/05	17:48	2500

64	02/01/05	17:50	3000
65	02/01/05	18:00	100
66	02/01/05	19:19	2000
67	02/01/05	19:24	800
68	02/01/05	19:28	500
69	02/01/05	19:58	300
70	02/01/05	20:31	1600
71	02/01/05	20:50	1000
72	02/01/05	20:53	100
73	02/01/05	21:08	2000
74	02/01/05	21:44	400
75	02/01/05	22:22	100
76	03/01/05	08:50	1500
77	03/01/05	10:03	6500
78	03/01/05	10:06	3500
79	03/01/05	10:08	1500
80	03/01/05	11:28	100
81	03/01/05	11:34	2000
82	03/01/05	11:35	300
83	03/01/05	11:45	500
84	03/01/05	12:00	1000
85	03/01/05	12:18	3000
86	03/01/05	12:55	1000
87	03/01/05	13:14	1600
88	03/01/05	13:25	100
89	03/01/05	13:27	100
90	03/01/05	13:28	100
91	03/01/05	13:43	400
92	03/01/05	13:44	3500
93	03/01/05	13:54	400
94	03/01/05	14:05	5000
95	03/01/05	14:13	5000
96	03/01/05	15:31	6500
97	03/01/05	16:11	100
98	03/01/05	16:13	600
99	03/01/05	16:15	3000
100	03/01/05	16:34	1500
101	03/01/05	17:15	4000
102	03/01/05	17:16	5500
103	03/01/05	17:22	1000
104	03/01/05	17:44	1300
105	03/01/05	17:50	13500
106	03/01/05	18:15	100
107	03/01/05	19:11	1000
108	03/01/05	19:20	5100
109	03/01/05	20:04	3000
110	03/01/05	20:27	500
111	03/01/05	22:31	20000
112	03/01/05	22:32	20000
113	03/01/05	23:45	100
114	04/01/05	10:51	600
115	04/01/05	11:22	1000
116	04/01/05	11:29	3000
117	04/01/05	11:31	10000
118	04/01/05	11:36	3000
119	04/01/05	12:03	500
120	04/01/05	12:29	200
121	04/01/05	12:35	13000
122	04/01/05	12:55	20000
123	04/01/05	13:30	20000
124	04/01/05	14:28	800
125	04/01/05	14:29	3000
126	04/01/05	14:30	2500
127	04/01/05	14:37	6500
128	04/01/05	15:37	1000
129	04/01/05	16:09	2000
130	04/01/05	16:14	8000
131	04/01/05	16:31	4100
132	04/01/05	16:35	2000
133	04/01/05	16:37	1300
134	04/01/05	17:26	20000
135	04/01/05	17:27	10500
136	04/01/05	17:39	500
137	04/01/05	18:06	500
138	04/01/05	18:07	20000
139	04/01/05	18:08	10000
140	04/01/05	18:10	1000
141	04/01/05	18:12	500
142	04/01/05	19:14	4000
143	04/01/05	19:18	10000
144	04/01/05	19:34	12500
145	04/01/05	19:40	500
146	04/01/05	21:14	100
147	04/01/05	22:47	200
148	04/01/05	23:41	3500
149	05/01/05	09:35	10000
150	05/01/05	09:54	3000
151	05/01/05	10:29	8500
152	05/01/05	10:59	100
153	05/01/05	11:01	2000

154	05/01/05	12:07	100
155	05/01/05	12:59	500
156	05/01/05	13:44	1000
157	05/01/05	13:45	1000
158	05/01/05	14:19	700
159	05/01/05	14:45	14000
160	05/01/05	15:02	20000
161	05/01/05	15:03	20000
162	05/01/05	15:34	5000
163	05/01/05	16:29	500
164	05/01/05	16:36	2000
165	05/01/05	17:08	500
166	05/01/05	18:00	10000
167	05/01/05	18:05	500
168	05/01/05	18:06	3000
169	05/01/05	18:08	100
170	05/01/05	18:16	3000
171	05/01/05	18:25	500
172	05/01/05	18:30	6000
173	05/01/05	19:34	10000
174	05/01/05	19:39	2000
175	05/01/05	19:44	10300
176	05/01/05	19:45	100
177	05/01/05	20:19	100
178	05/01/05	20:20	100
179	05/01/05	20:59	2500
180	05/01/05	22:38	3000
181	06/01/05	09:39	8500
182	06/01/05	09:40	500
183	06/01/05	09:45	4500
184	06/01/05	09:45	4500
185	06/01/05	11:29	10000
186	06/01/05	11:33	200
187	06/01/05	11:51	100
188	06/01/05	12:43	500
189	06/01/05	12:57	1100
190	06/01/05	13:01	200
191	06/01/05	13:22	200
192	06/01/05	13:24	5000
193	06/01/05	13:26	13000
194	06/01/05	13:37	500
195	06/01/05	14:45	20000
196	06/01/05	15:06	3000
197	06/01/05	15:31	1000
198	06/01/05	15:35	20000
199	06/01/05	16:34	3100
200	06/01/05	16:37	4000
201	06/01/05	17:07	10000
202	06/01/05	17:22	200
203	06/01/05	18:26	3000
204	06/01/05	18:30	3000
205	06/01/05	18:31	2000
206	06/01/05	18:41	500
207	06/01/05	18:48	300
208	06/01/05	19:00	1100
209	06/01/05	19:15	1000
210	06/01/05	19:27	20000
211	06/01/05	20:08	100
212	06/01/05	21:58	1000
213	06/01/05	22:01	10000
214	07/01/05	10:55	4300
215	07/01/05	10:56	6000
216	07/01/05	10:57	300
217	07/01/05	10:59	10000
218	07/01/05	11:21	6000
219	07/01/05	11:34	300
220	07/01/05	11:45	3000
221	07/01/05	11:45	2000
222	07/01/05	11:47	500
223	07/01/05	11:50	600
224	07/01/05	12:02	600
225	07/01/05	12:14	1000
226	07/01/05	12:31	15000
227	07/01/05	12:34	1200
228	07/01/05	12:55	2000
229	07/01/05	13:34	10000
230	07/01/05	14:16	7000
231	07/01/05	14:19	20000
232	07/01/05	15:49	500
233	07/01/05	18:14	400
234	07/01/05	18:16	9000
235	07/01/05	18:27	600
236	07/01/05	18:50	300
237	07/01/05	18:57	3600
238	07/01/05	19:09	2900
239	07/01/05	19:36	100
240	07/01/05	19:51	200
241	07/01/05	20:00	700
242	07/01/05	21:44	1000
243	08/01/05	11:04	20000

244	08/01/05	11:59	10000
245	08/01/05	12:25	20000
246	08/01/05	12:26	20000
247	08/01/05	12:28	5000
248	08/01/05	13:03	500
249	08/01/05	13:16	100
250	08/01/05	13:36	10000
251	08/01/05	13:42	3000
252	08/01/05	13:45	3000
253	08/01/05	13:54	2000
254	08/01/05	13:55	2000
255	08/01/05	15:19	5000
256	08/01/05	15:21	1000
257	08/01/05	16:00	1000
258	08/01/05	16:04	2200
259	08/01/05	16:43	5000
260	08/01/05	16:53	6000
261	08/01/05	17:08	20000
262	08/01/05	17:09	10000
263	08/01/05	17:28	3000
264	08/01/05	18:53	700
265	08/01/05	19:08	2200
266	08/01/05	19:11	2000
267	08/01/05	19:17	1000
268	08/01/05	19:59	2000
269	08/01/05	20:00	20000
270	08/01/05	20:01	18000
271	08/01/05	20:04	2000
272	08/01/05	20:16	500
273	08/01/05	20:27	1000
274	08/01/05	20:46	3000
275	09/01/05	07:43	1000
276	09/01/05	09:50	500
277	09/01/05	10:30	20000
278	09/01/05	10:32	300
279	09/01/05	10:42	500
280	09/01/05	10:51	500
281	09/01/05	10:52	3000
282	09/01/05	10:54	3000
283	09/01/05	10:55	500
284	09/01/05	11:52	500
285	09/01/05	12:50	1000
286	09/01/05	12:55	500
287	09/01/05	12:58	10000
288	09/01/05	13:01	100
289	09/01/05	13:04	7000
290	09/01/05	15:35	300
291	09/01/05	16:30	20000
292	09/01/05	17:49	3000
293	09/01/05	17:52	100
294	09/01/05	19:03	200
295	09/01/05	20:26	700
296	09/01/05	20:29	3000
297	09/01/05	20:41	100
298	09/01/05	20:44	500
299	09/01/05	20:50	1000
300	10/01/05	10:34	300
301	10/01/05	10:53	1500
302	10/01/05	10:54	15000
303	10/01/05	11:08	1500
304	10/01/05	11:30	1500
305	10/01/05	11:47	100
306	10/01/05	11:57	3000
307	10/01/05	12:30	800
308	10/01/05	12:37	4500
309	10/01/05	12:54	500
310	10/01/05	12:57	10000
311	10/01/05	12:57	10000
312	10/01/05	13:19	2000
313	10/01/05	13:46	12500
314	10/01/05	13:55	6000
315	10/01/05	14:30	1000
316	10/01/05	14:34	2000
317	10/01/05	15:42	200
318	10/01/05	15:47	2000
319	10/01/05	17:14	7000
320	10/01/05	17:27	5000
321	10/01/05	17:33	9000
322	10/01/05	17:43	500
323	10/01/05	17:44	100
324	10/01/05	17:56	500
325	10/01/05	17:57	1000
326	10/01/05	18:13	8000
327	10/01/05	18:50	1000
328	10/01/05	18:55	500
329	11/01/05	10:35	5000
330	11/01/05	11:02	6000
331	11/01/05	11:40	1800
332	11/01/05	12:05	2500
333	11/01/05	12:09	800

334	11/01/05	12:13	1000
335	11/01/05	12:14	100
336	11/01/05	12:17	3000
337	11/01/05	12:25	13000
338	11/01/05	12:37	7000
339	11/01/05	12:56	11000
340	11/01/05	12:57	4000
341	11/01/05	13:00	500
342	11/01/05	13:11	800
343	11/01/05	13:18	20000
344	11/01/05	13:19	10000
345	11/01/05	13:40	500
346	11/01/05	14:02	3000
347	11/01/05	14:17	300
348	11/01/05	15:06	300
349	11/01/05	15:13	10000
350	11/01/05	15:39	10000
351	11/01/05	15:40	5000
352	11/01/05	15:41	5000
353	11/01/05	16:26	1000
354	11/01/05	16:57	500
355	11/01/05	17:06	2500
356	11/01/05	17:57	1000
357	11/01/05	18:16	1500
358	11/01/05	18:39	1200
359	11/01/05	18:50	600
360	11/01/05	18:55	3000
361	11/01/05	18:58	100
362	11/01/05	19:24	3000
363	11/01/05	21:37	5000
364	11/01/05	22:09	3000
365	12/01/05	09:07	500
366	12/01/05	09:12	300
367	12/01/05	10:09	100
368	12/01/05	10:30	300
369	12/01/05	10:57	300
370	12/01/05	11:23	5000
371	12/01/05	11:26	10000
372	12/01/05	11:37	500
373	12/01/05	11:38	500
374	12/01/05	11:39	500
375	12/01/05	11:40	500
376	12/01/05	11:41	500
377	12/01/05	11:42	500
378	12/01/05	11:45	500
379	12/01/05	11:48	1000
380	12/01/05	11:58	2000
381	12/01/05	12:28	20000
382	12/01/05	12:38	100
383	12/01/05	12:46	3700
384	12/01/05	14:09	400
385	12/01/05	14:37	3000
386	12/01/05	14:53	20000
387	12/01/05	14:54	5000
388	12/01/05	14:56	1000
389	12/01/05	15:01	3000
390	12/01/05	15:12	700
391	12/01/05	15:15	8000
392	12/01/05	15:30	18000
393	12/01/05	15:57	3000
394	12/01/05	16:33	100
395	12/01/05	16:36	100
396	12/01/05	16:58	500
397	12/01/05	17:04	10000
398	12/01/05	17:21	12000
399	12/01/05	17:26	200
400	12/01/05	17:30	3000
401	12/01/05	18:14	500
402	12/01/05	19:08	200
403	12/01/05	19:41	2500
404	12/01/05	20:07	800
405	12/01/05	20:32	1000
406	12/01/05	20:33	500
407	12/01/05	21:21	4000
408	12/01/05	21:51	6100
409	12/01/05	22:31	1000
410	13/01/05	08:24	700
411	13/01/05	10:23	10000
412	13/01/05	11:11	17000
413	13/01/05	11:21	8000
414	13/01/05	11:30	10000
415	13/01/05	11:46	500
416	13/01/05	11:50	1000
417	13/01/05	11:54	500
418	13/01/05	11:57	500
419	13/01/05	12:11	600
420	13/01/05	12:12	20000
421	13/01/05	12:13	10000
422	13/01/05	12:19	20000
423	13/01/05	12:24	10000

424	13/01/05	12:38	14000
425	13/01/05	12:41	1000
426	13/01/05	12:44	3000
427	13/01/05	13:00	500
428	13/01/05	13:18	3000
429	13/01/05	13:33	3000
430	13/01/05	13:38	700
431	13/01/05	13:41	100
432	13/01/05	13:57	100
433	13/01/05	15:13	20000
434	13/01/05	15:14	10000
435	13/01/05	15:19	1500
436	13/01/05	15:44	2000
437	13/01/05	16:20	3000
438	13/01/05	17:43	500
439	13/01/05	17:46	1000
440	13/01/05	17:54	500
441	13/01/05	17:57	500
442	13/01/05	18:12	3500
443	13/01/05	18:27	3000
444	13/01/05	18:27	3000
445	13/01/05	18:35	10000
446	13/01/05	18:37	300
447	13/01/05	19:01	500
448	13/01/05	19:13	100
449	13/01/05	19:35	200
450	13/01/05	19:54	1000
451	13/01/05	20:04	100
452	13/01/05	20:44	100
453	14/01/05	09:45	400
454	14/01/05	10:11	2700
455	14/01/05	10:46	1000
456	14/01/05	11:35	1000
457	14/01/05	12:33	5000
458	14/01/05	12:49	2500
459	14/01/05	13:07	200
460	14/01/05	13:09	300
461	14/01/05	13:26	2000
462	14/01/05	13:30	500
463	14/01/05	13:59	3000
464	14/01/05	14:01	3000
465	14/01/05	14:02	1000
466	14/01/05	15:01	600
467	14/01/05	15:05	300
468	14/01/05	16:10	500
469	14/01/05	16:16	300
470	14/01/05	16:27	200
471	14/01/05	19:37	5000
472	14/01/05	21:00	1000
473	14/01/05	21:03	100
474	14/01/05	21:33	200
475	14/01/05	23:03	5000
476	15/01/05	08:57	2800
477	15/01/05	08:58	5000
478	15/01/05	09:50	5000
479	15/01/05	09:59	300
480	15/01/05	10:00	10000
481	15/01/05	10:14	2000
482	15/01/05	11:29	1500
483	15/01/05	11:32	5000
484	15/01/05	11:49	5000
485	15/01/05	12:46	200
486	15/01/05	13:08	18000
487	15/01/05	13:23	300
488	15/01/05	13:49	5500
489	15/01/05	14:29	15000
490	15/01/05	15:18	900
491	15/01/05	16:12	700
492	15/01/05	16:29	300
493	15/01/05	17:22	1500
494	15/01/05	17:25	2000
495	15/01/05	17:27	300
496	15/01/05	17:50	600
497	15/01/05	18:06	200
498	15/01/05	18:09	20000
499	15/01/05	18:10	5000
500	15/01/05	18:31	200
501	15/01/05	18:59	500
502	15/01/05	19:45	3000
503	15/01/05	19:59	600
504	15/01/05	20:14	2500
505	15/01/05	21:46	3600
506	16/01/05	10:31	2000
507	16/01/05	10:35	3000
508	16/01/05	11:06	1000
509	16/01/05	11:08	300
510	16/01/05	11:56	1500
511	16/01/05	11:59	1000
512	16/01/05	12:21	1000
513	16/01/05	13:07	100

514	16/01/05	13:30	2000
515	16/01/05	17:30	1000
516	16/01/05	18:34	2000
517	16/01/05	18:47	1000
518	16/01/05	19:02	500
519	16/01/05	19:22	500
520	16/01/05	21:55	100
521	17/01/05	09:12	2700
522	17/01/05	09:37	100
523	17/01/05	10:55	100
524	17/01/05	11:12	500
525	17/01/05	11:40	20000
526	17/01/05	11:52	2000
527	17/01/05	11:59	1000
528	17/01/05	12:04	18000
529	17/01/05	12:05	1000
530	17/01/05	12:19	500
531	17/01/05	12:58	500
532	17/01/05	13:57	400
533	17/01/05	14:20	2500
534	17/01/05	15:06	20000
535	17/01/05	15:08	6000
536	17/01/05	15:10	10000
537	17/01/05	15:11	200
538	17/01/05	15:48	20000
539	17/01/05	16:35	1000
540	17/01/05	17:03	2500
541	17/01/05	17:30	10000
542	17/01/05	17:32	10500
543	17/01/05	17:52	20000
544	17/01/05	17:54	14500
545	17/01/05	18:13	1000
546	17/01/05	18:14	300
547	17/01/05	18:40	600
548	17/01/05	18:43	2000
549	17/01/05	19:08	100
550	17/01/05	19:11	500
551	17/01/05	19:19	1000
552	17/01/05	19:25	600
553	17/01/05	20:19	1000
554	17/01/05	20:21	1000
555	17/01/05	20:23	1000
556	17/01/05	20:25	1000
557	17/01/05	20:32	1000
558	17/01/05	20:36	8000
559	18/01/05	09:05	3000
560	18/01/05	09:06	3000
561	18/01/05	09:14	2000
562	18/01/05	10:11	2000
563	18/01/05	10:28	2400
564	18/01/05	13:15	1000
565	18/01/05	13:21	200
566	18/01/05	13:36	15000
567	18/01/05	14:08	2500
568	18/01/05	14:10	500
569	18/01/05	14:11	3000
570	18/01/05	14:18	4500
571	18/01/05	14:26	3000
572	18/01/05	14:39	10500
573	18/01/05	15:44	100
574	18/01/05	15:45	100
575	18/01/05	16:07	10000
576	18/01/05	16:18	3000
577	18/01/05	16:21	3000
578	18/01/05	16:22	3000
579	18/01/05	16:23	3000
580	18/01/05	16:24	3000
581	18/01/05	16:25	3000
582	18/01/05	16:26	1000
583	18/01/05	16:36	1500
584	18/01/05	16:40	3000
585	18/01/05	16:41	2000
586	18/01/05	16:50	2000
587	18/01/05	17:29	10000
588	18/01/05	17:46	400
589	18/01/05	17:56	500
590	18/01/05	18:27	6000
591	18/01/05	18:30	300
592	18/01/05	18:39	100
593	18/01/05	21:15	500
594	19/01/05	09:56	100
595	19/01/05	10:45	500
596	19/01/05	11:23	2000
597	19/01/05	11:27	11000
598	19/01/05	11:53	21000
599	19/01/05	12:03	1000
600	19/01/05	12:10	1000
601	19/01/05	12:16	1300
602	19/01/05	12:18	2000
603	19/01/05	12:20	200

604	19/01/05	12:34	100
605	19/01/05	13:05	500
606	19/01/05	13:25	1000
607	19/01/05	13:26	1000
608	19/01/05	13:26	500
609	19/01/05	13:34	3000
610	19/01/05	13:36	100
611	19/01/05	13:45	1000
612	19/01/05	13:53	1000
613	19/01/05	14:01	4900
614	19/01/05	14:03	500
615	19/01/05	14:35	100
616	19/01/05	16:19	400
617	19/01/05	16:28	300
618	19/01/05	16:33	3000
619	19/01/05	17:12	100
620	19/01/05	17:35	500
621	19/01/05	17:36	200
622	19/01/05	19:21	100
623	19/01/05	19:34	20000
624	19/01/05	19:35	20000
625	19/01/05	19:45	3000
626	19/01/05	20:14	1000
627	19/01/05	21:00	400
628	19/01/05	21:36	200
629	19/01/05	21:39	500
630	20/01/05	07:44	100
631	20/01/05	10:37	2000
632	20/01/05	10:43	7000
633	20/01/05	11:31	200
634	20/01/05	14:23	2000
635	20/01/05	14:35	600
636	20/01/05	14:36	500
637	20/01/05	15:34	1800
638	20/01/05	15:35	200
639	20/01/05	16:01	6000
640	20/01/05	16:05	100
641	20/01/05	16:13	1500
642	20/01/05	16:18	1000
643	20/01/05	16:52	10000
644	20/01/05	16:54	25000
645	20/01/05	17:04	2000
646	20/01/05	17:51	5000
647	20/01/05	17:55	14000
648	20/01/05	18:00	300
649	20/01/05	18:05	300
650	20/01/05	18:12	1000
651	20/01/05	18:20	5100
652	20/01/05	18:22	3000
653	20/01/05	18:48	500
654	20/01/05	18:50	500
655	20/01/05	19:43	300
656	20/01/05	19:50	500
657	20/01/05	20:53	500
658	20/01/05	21:39	500
659	20/01/05	21:42	300
660	20/01/05	21:43	4100
661	21/01/05	08:24	500
662	21/01/05	09:10	500
663	21/01/05	10:15	6000
664	21/01/05	11:10	500
665	21/01/05	11:31	500
666	21/01/05	11:36	800
667	21/01/05	11:38	1000
668	21/01/05	11:54	12000
669	21/01/05	12:10	3000
670	21/01/05	12:54	2000
671	21/01/05	13:00	300
672	21/01/05	13:17	2000
673	21/01/05	14:28	3000
674	21/01/05	14:37	100
675	21/01/05	15:40	1500
676	21/01/05	16:37	500
677	21/01/05	17:41	500
678	21/01/05	17:43	1000
679	21/01/05	18:15	1800
680	21/01/05	18:51	1000
681	21/01/05	18:58	2000
682	21/01/05	18:59	100
683	21/01/05	19:29	100
684	21/01/05	19:43	500
685	21/01/05	19:59	500
686	21/01/05	20:04	500
687	21/01/05	20:43	1500
688	21/01/05	23:25	10000
689	22/01/05	08:44	1000
690	22/01/05	08:46	1000
691	22/01/05	10:09	5000
692	22/01/05	10:40	200
693	22/01/05	11:34	6000

694	22/01/05	11:40	500
695	22/01/05	11:51	5000
696	22/01/05	12:47	100
697	22/01/05	12:55	300
698	22/01/05	12:58	200
699	22/01/05	13:37	1500
700	22/01/05	13:42	500
701	22/01/05	14:01	500
702	22/01/05	16:43	10000
703	22/01/05	16:52	2000
704	22/01/05	17:21	4400
705	22/01/05	17:36	6000
706	22/01/05	17:42	1000
707	22/01/05	17:44	20000
708	22/01/05	17:49	3000
709	22/01/05	18:39	1000
710	22/01/05	18:52	9000
711	22/01/05	19:08	300
712	22/01/05	19:14	200
713	22/01/05	20:03	5000
714	22/01/05	20:25	8600
715	22/01/05	21:02	3000
716	22/01/05	21:06	6000
717	23/01/05	07:59	500
718	23/01/05	11:44	11000
719	23/01/05	13:32	800
720	23/01/05	13:33	100
721	23/01/05	14:12	500
722	23/01/05	14:31	500
723	23/01/05	17:17	500
724	23/01/05	17:22	100
725	23/01/05	17:41	200
726	23/01/05	17:42	10000
727	23/01/05	17:58	1000
728	23/01/05	17:59	900
729	23/01/05	18:02	500
730	23/01/05	18:05	1500
731	23/01/05	18:06	500
732	23/01/05	18:07	3000
733	23/01/05	18:48	500
734	23/01/05	18:53	1000
735	23/01/05	19:06	500
736	23/01/05	19:44	700
737	23/01/05	19:47	300
738	23/01/05	20:02	2500
739	24/01/05	10:27	200
740	24/01/05	10:49	20000
741	24/01/05	10:50	20000
742	24/01/05	11:02	18000
743	24/01/05	11:11	100
744	24/01/05	11:13	10000
745	24/01/05	11:38	300
746	24/01/05	11:39	1000
747	24/01/05	11:40	100
748	24/01/05	11:44	100
749	24/01/05	11:52	200
750	24/01/05	12:07	5000
751	24/01/05	12:11	10000
752	24/01/05	12:18	4000
753	24/01/05	13:46	2300
754	24/01/05	15:15	20000
755	24/01/05	15:16	10800
756	24/01/05	15:58	1000
757	24/01/05	18:03	100
758	24/01/05	18:27	500
759	24/01/05	18:29	900
760	24/01/05	18:36	200
761	24/01/05	18:51	1500
762	24/01/05	18:55	300
763	24/01/05	19:44	100
764	24/01/05	20:38	10000
765	24/01/05	20:56	700
766	25/01/05	07:56	100
767	25/01/05	09:25	100
768	25/01/05	09:36	3000
769	25/01/05	09:44	100
770	25/01/05	09:46	500
771	25/01/05	10:03	5000
772	25/01/05	10:37	700
773	25/01/05	10:41	1500
774	25/01/05	10:46	3000
775	25/01/05	10:47	10000
776	25/01/05	10:48	7000
777	25/01/05	10:59	3000
778	25/01/05	11:19	15000
779	25/01/05	11:20	1000
780	25/01/05	11:30	500
781	25/01/05	11:37	300
782	25/01/05	11:38	100
783	25/01/05	11:41	2300

784	25/01/05	12:17	10000
785	25/01/05	12:33	1500
786	25/01/05	13:10	10500
787	25/01/05	13:14	500
788	25/01/05	13:24	20000
789	25/01/05	13:55	1000
790	25/01/05	14:00	10000
791	25/01/05	15:18	2000
792	25/01/05	15:20	500
793	25/01/05	15:35	8800
794	25/01/05	15:37	2000
795	25/01/05	16:32	200
796	25/01/05	16:40	1700
797	25/01/05	17:19	300
798	25/01/05	17:22	200
799	25/01/05	17:25	3000
800	25/01/05	17:33	200
801	25/01/05	17:40	100
802	25/01/05	17:42	17000
803	25/01/05	17:49	2000
804	25/01/05	18:25	2000
805	25/01/05	18:47	200
806	25/01/05	19:10	100
807	25/01/05	19:24	1000
808	25/01/05	19:27	500
809	25/01/05	19:41	200
810	25/01/05	19:47	9900
811	25/01/05	19:49	14000
812	25/01/05	19:59	5000
813	25/01/05	20:05	300
814	25/01/05	20:11	100
815	25/01/05	20:27	200
816	25/01/05	20:32	400
817	25/01/05	20:40	100
818	25/01/05	20:52	500
819	25/01/05	21:48	1500
820	25/01/05	23:22	2000
821	25/01/05	23:25	1000
822	26/01/05	00:39	300
823	26/01/05	06:53	3200
824	26/01/05	09:39	200
825	26/01/05	11:20	400
826	26/01/05	11:46	1500
827	26/01/05	11:48	2000
828	26/01/05	11:51	500
829	26/01/05	11:55	500
830	26/01/05	12:00	4000
831	26/01/05	12:11	500
832	26/01/05	12:36	500
833	26/01/05	12:38	600
834	26/01/05	12:41	10400
835	26/01/05	13:17	3500
836	26/01/05	13:39	8000
837	26/01/05	13:57	20000
838	26/01/05	13:58	5000
839	26/01/05	14:04	500
840	26/01/05	14:25	2000
841	26/01/05	15:31	2000
842	26/01/05	15:50	20000
843	26/01/05	15:55	10000
844	26/01/05	17:39	1000
845	26/01/05	17:41	1500
846	26/01/05	18:52	5000
847	26/01/05	20:00	100
848	26/01/05	21:00	1000
849	26/01/05	22:30	200
850	26/01/05	23:31	4000
851	27/01/05	09:23	100
852	27/01/05	10:34	20000
853	27/01/05	10:36	5000
854	27/01/05	11:32	1200
855	27/01/05	11:43	600
856	27/01/05	11:44	400
857	27/01/05	11:45	1000
858	27/01/05	11:46	10000
859	27/01/05	11:51	5000
860	27/01/05	11:57	20000
861	27/01/05	12:13	1000
862	27/01/05	12:13	3000
863	27/01/05	12:38	400
864	27/01/05	12:54	7000
865	27/01/05	13:07	1500
866	27/01/05	13:09	2000
867	27/01/05	13:11	700
868	27/01/05	13:21	500
869	27/01/05	13:22	300
870	27/01/05	13:40	100
871	27/01/05	14:40	20000
872	27/01/05	14:41	20000
873	27/01/05	15:28	500

874	27/01/05	15:36	400
875	27/01/05	15:51	100
876	27/01/05	15:54	15000
877	27/01/05	16:01	3000
878	27/01/05	16:13	200
879	27/01/05	16:55	5000
880	27/01/05	17:06	200
881	27/01/05	17:15	3000
882	27/01/05	18:11	4000
883	27/01/05	18:14	200
884	27/01/05	18:15	300
885	27/01/05	19:08	100
886	27/01/05	19:29	100
887	27/01/05	19:44	500
888	27/01/05	20:19	500
889	27/01/05	23:48	6000
890	28/01/05	10:01	5000
891	28/01/05	10:04	300
892	28/01/05	10:36	1000
893	28/01/05	10:46	12000
894	28/01/05	11:25	500
895	28/01/05	11:54	6500
896	28/01/05	12:02	10000
897	28/01/05	12:22	8000
898	28/01/05	12:31	5000
899	28/01/05	13:14	100
900	28/01/05	13:34	100
901	28/01/05	13:37	100
902	28/01/05	13:50	100
903	28/01/05	15:33	2500
904	28/01/05	15:35	2700
905	28/01/05	16:36	1000
906	28/01/05	16:41	5500
907	28/01/05	17:12	500
908	28/01/05	17:20	1600
909	28/01/05	17:26	300
910	28/01/05	17:37	15000
911	28/01/05	17:42	500
912	28/01/05	17:59	1800
913	28/01/05	18:02	500
914	28/01/05	19:04	500
915	28/01/05	19:06	5500
916	28/01/05	19:13	100
917	28/01/05	19:20	100
918	28/01/05	19:23	200
919	28/01/05	20:10	4000
920	28/01/05	21:29	15000
921	29/01/05	06:32	500
922	29/01/05	10:14	1500
923	29/01/05	10:15	700
924	29/01/05	10:32	7000
925	29/01/05	10:35	1000
926	29/01/05	10:38	100
927	29/01/05	10:52	1500
928	29/01/05	10:54	3000
929	29/01/05	11:16	1000
930	29/01/05	11:19	1000
931	29/01/05	11:49	1000
932	29/01/05	11:53	3000
933	29/01/05	12:01	13500
934	29/01/05	12:19	11000
935	29/01/05	12:25	7800
936	29/01/05	12:50	9500
937	29/01/05	13:21	100
938	29/01/05	13:23	20000
939	29/01/05	13:24	10000
940	29/01/05	14:33	100
941	29/01/05	14:42	200
942	29/01/05	14:52	15000
943	29/01/05	16:26	20000
944	29/01/05	16:31	400
945	29/01/05	16:59	3000
946	29/01/05	17:53	6500
947	29/01/05	17:55	100
948	29/01/05	18:36	4100
949	29/01/05	18:53	1000
950	29/01/05	18:54	500
951	29/01/05	19:05	200
952	29/01/05	19:06	200
953	29/01/05	19:21	100
954	29/01/05	19:23	500
955	29/01/05	19:33	300
956	29/01/05	20:00	200
957	29/01/05	20:29	500
958	29/01/05	20:48	2000
959	29/01/05	20:50	3000
960	29/01/05	21:03	2500
961	29/01/05	22:16	10000
962	30/01/05	11:05	200
963	30/01/05	11:33	14000

964	30/01/05	11:50	200
965	30/01/05	11:52	500
966	30/01/05	11:53	200
967	30/01/05	12:45	200
968	30/01/05	12:47	9000
969	30/01/05	12:50	100
970	30/01/05	12:51	1000
971	30/01/05	12:56	1400
972	30/01/05	15:05	1500
973	30/01/05	15:21	100
974	30/01/05	15:47	1000
975	30/01/05	16:41	500
976	30/01/05	17:36	1000
977	30/01/05	18:35	300
978	30/01/05	18:44	1500
979	30/01/05	18:48	300
980	30/01/05	19:06	100
981	30/01/05	19:08	200
982	30/01/05	20:05	100
983	30/01/05	20:26	100
984	30/01/05	21:11	400
985	30/01/05	22:27	10000
986	30/01/05	23:35	5000
987	30/01/05	23:52	5000
988	31/01/05	08:55	100
989	31/01/05	09:27	500
990	31/01/05	09:35	2000
991	31/01/05	09:48	800
992	31/01/05	09:51	2500
993	31/01/05	09:53	600
994	31/01/05	09:58	20000
995	31/01/05	10:29	500
996	31/01/05	10:33	500
997	31/01/05	10:50	10000
998	31/01/05	11:02	2500
999	31/01/05	11:03	5900
1000	31/01/05	11:29	100
1001	31/01/05	11:30	200
1002	31/01/05	11:53	20000
1003	31/01/05	12:01	2800
1004	31/01/05	12:04	10000
1005	31/01/05	12:09	10000
1006	31/01/05	12:20	1000
1007	31/01/05	13:04	20000
1008	31/01/05	14:09	4700
1009	31/01/05	14:28	2000
1010	31/01/05	14:39	2000
1011	31/01/05	15:25	2000
1012	31/01/05	15:35	15000
1013	31/01/05	15:43	2000
1014	31/01/05	16:08	2000
1015	31/01/05	16:24	1000
1016	31/01/05	16:41	1000
1017	31/01/05	16:42	20000
1018	31/01/05	16:43	20000
1019	31/01/05	16:48	200
1020	31/01/05	16:49	20000
1021	31/01/05	16:52	20800
1022	31/01/05	16:53	5200
1023	31/01/05	17:22	5400
1024	31/01/05	17:34	100
1025	31/01/05	17:53	900
1026	31/01/05	18:14	300
1027	31/01/05	18:16	300
1028	31/01/05	18:20	5000
1029	31/01/05	18:21	1000
1030	31/01/05	18:50	300
1031	31/01/05	19:02	400
1032	31/01/05	19:10	600
1033	31/01/05	19:11	800
1034	31/01/05	19:43	5000
1035	31/01/05	20:01	5000
1036	31/01/05	20:11	200
1037	31/01/05	20:36	2500
1038	31/01/05	21:49	100
1039	31/01/05	21:54	700

.
.
.

December-2005

16169	01/12/05	08:18	20000	16215	02/12/05	18:48	200
16170	01/12/05	08:19	20000	16216	02/12/05	18:50	500
16171	01/12/05	09:40	3000	16217	02/12/05	18:53	3200
16172	01/12/05	09:43	3000	16218	02/12/05	18:55	3700
16173	01/12/05	09:51	10000	16219	02/12/05	20:33	500
16174	01/12/05	10:04	1000	16220	02/12/05	20:58	5500
16175	01/12/05	10:17	2000	16221	02/12/05	21:16	5900
16176	01/12/05	10:21	200	16222	02/12/05	21:39	1000
16177	01/12/05	10:24	200	16223	02/12/05	21:44	1000
16178	01/12/05	10:26	3000	16224	02/12/05	21:48	8000
16179	01/12/05	10:59	3000	16225	02/12/05	22:16	1000
16180	01/12/05	11:28	2000	16226	02/12/05	22:30	500
16181	01/12/05	11:30	300	16227	03/12/05	09:03	1000
16182	01/12/05	11:46	5000	16228	03/12/05	09:25	300
16183	01/12/05	11:59	200	16229	03/12/05	10:04	100
16184	01/12/05	12:10	2000	16230	03/12/05	10:39	300
16185	01/12/05	12:12	500	16231	03/12/05	10:45	11000
16186	01/12/05	12:13	500	16232	03/12/05	10:59	200
16187	01/12/05	15:54	5000	16233	03/12/05	11:01	6100
16188	01/12/05	16:05	500	16234	03/12/05	11:20	800
16189	01/12/05	17:06	600	16235	03/12/05	11:40	10000
16190	01/12/05	17:09	15000	16236	03/12/05	11:44	5000
16191	01/12/05	17:15	9500	16237	03/12/05	11:50	5000
16192	01/12/05	19:27	700	16238	03/12/05	12:58	5000
16193	01/12/05	19:30	5000	16239	03/12/05	13:00	100
16194	01/12/05	19:36	400	16240	03/12/05	13:41	3300
16195	01/12/05	19:59	800	16241	03/12/05	13:43	1000
16196	02/12/05	07:39	5000	16242	03/12/05	13:54	500
16197	02/12/05	10:01	500	16243	03/12/05	14:00	8000
16198	02/12/05	10:03	200	16244	03/12/05	14:06	4500
16199	02/12/05	10:07	300	16245	03/12/05	14:07	2500
16200	02/12/05	10:25	300	16246	03/12/05	14:51	1000
16201	02/12/05	10:58	1000	16247	03/12/05	14:57	10000
16202	02/12/05	11:46	5000	16248	03/12/05	15:30	15000
16203	02/12/05	12:05	1500	16249	03/12/05	15:42	1000
16204	02/12/05	12:17	500	16250	03/12/05	16:00	2000
16205	02/12/05	12:39	100	16251	03/12/05	16:11	1000
16206	02/12/05	12:52	1000	16252	03/12/05	16:29	2000
16207	02/12/05	12:53	500	16253	03/12/05	17:02	2500
16208	02/12/05	13:11	500	16254	03/12/05	17:32	1000
16209	02/12/05	13:39	300	16255	03/12/05	17:36	2300
16210	02/12/05	13:45	1000	16256	03/12/05	18:18	4900
16211	02/12/05	16:09	2000	16257	03/12/05	18:19	300
16212	02/12/05	16:46	200	16258	03/12/05	18:38	5000
16213	02/12/05	18:27	500	16259	03/12/05	18:59	3500
16214	02/12/05	18:39	200	16260	03/12/05	19:03	3000

16261	03/12/05	19:42	500	16309	04/12/05	17:09	5000
16262	03/12/05	19:45	3000	16310	04/12/05	17:11	3000
16263	03/12/05	19:48	1500	16311	04/12/05	17:29	2500
16264	03/12/05	20:07	200	16312	04/12/05	17:37	5000
16265	03/12/05	20:12	500	16313	04/12/05	17:51	1000
16266	03/12/05	20:12	500	16314	04/12/05	18:54	5000
16267	03/12/05	20:19	300	16315	04/12/05	18:55	1700
16268	03/12/05	20:28	1500	16316	04/12/05	18:56	5000
16269	03/12/05	20:37	3000	16317	04/12/05	18:58	100
16270	03/12/05	20:38	2000	16318	04/12/05	19:04	4700
16271	03/12/05	21:08	300	16319	04/12/05	19:10	1000
16272	03/12/05	21:20	200	16320	04/12/05	20:29	1000
16273	03/12/05	21:33	3000	16321	04/12/05	20:42	200
16274	03/12/05	21:41	100	16322	04/12/05	21:06	100
16275	03/12/05	23:00	1000	16323	04/12/05	21:52	700
16276	04/12/05	10:00	2000	16324	04/12/05	22:39	100
16277	04/12/05	10:22	2500	16325	04/12/05	22:53	200
16278	04/12/05	10:28	3000	16326	04/12/05	23:05	300
16279	04/12/05	11:36	500	16327	05/12/05	00:03	2000
16280	04/12/05	11:38	500	16328	05/12/05	07:58	1000
16281	04/12/05	11:41	2000	16329	05/12/05	07:59	500
16282	04/12/05	12:01	200	16330	05/12/05	09:31	500
16283	04/12/05	12:33	2000	16331	05/12/05	10:26	10000
16284	04/12/05	12:40	5000	16332	05/12/05	10:55	3000
16285	04/12/05	12:52	100	16333	05/12/05	10:57	1800
16286	04/12/05	13:17	1000	16334	05/12/05	11:07	200
16287	04/12/05	13:19	5000	16335	05/12/05	11:51	3000
16288	04/12/05	13:26	2000	16336	05/12/05	11:53	300
16289	04/12/05	13:28	2000	16337	05/12/05	11:54	1000
16290	04/12/05	13:31	10000	16338	05/12/05	11:57	1000
16291	04/12/05	13:38	100	16339	05/12/05	12:13	3000
16292	04/12/05	13:43	2200	16340	05/12/05	12:25	200
16293	04/12/05	14:26	1200	16341	05/12/05	12:41	2000
16294	04/12/05	14:30	800	16342	05/12/05	12:47	1000
16295	04/12/05	14:32	3000	16343	05/12/05	12:51	3000
16296	04/12/05	14:33	1000	16344	05/12/05	13:41	3000
16297	04/12/05	15:00	20000	16345	05/12/05	13:56	10000
16298	04/12/05	15:06	200	16346	05/12/05	13:57	1000
16299	04/12/05	15:13	5000	16347	05/12/05	14:31	1000
16300	04/12/05	15:19	1200	16348	05/12/05	14:54	5000
16301	04/12/05	15:29	1100	16349	05/12/05	15:19	6500
16302	04/12/05	15:47	7000	16350	05/12/05	15:22	3000
16303	04/12/05	15:53	7000	16351	05/12/05	15:23	3000
16304	04/12/05	16:24	3000	16352	05/12/05	15:29	300
16305	04/12/05	16:31	11000	16353	05/12/05	16:05	2000
16306	04/12/05	16:33	3000	16354	05/12/05	16:41	5000
16307	04/12/05	16:35	200	16355	05/12/05	16:55	1000
16308	04/12/05	16:38	4300	16356	05/12/05	17:08	500

16357	05/12/05	17:25	100	16405	06/12/05	18:39	100
16358	05/12/05	17:57	3000	16406	06/12/05	18:43	2000
16359	05/12/05	18:23	700	16407	06/12/05	18:53	1700
16360	05/12/05	18:26	800	16408	06/12/05	19:02	900
16361	05/12/05	18:27	200	16409	06/12/05	19:06	500
16362	05/12/05	18:47	2500	16410	06/12/05	19:13	2200
16363	05/12/05	19:05	500	16411	06/12/05	19:36	2000
16364	05/12/05	19:06	1000	16412	06/12/05	19:38	5000
16365	05/12/05	19:10	3000	16413	06/12/05	20:19	2000
16366	05/12/05	19:13	3500	16414	06/12/05	20:33	10000
16367	05/12/05	20:39	9000	16415	06/12/05	20:50	200
16368	05/12/05	20:43	300	16416	06/12/05	21:19	200
16369	06/12/05	12:06	500	16417	06/12/05	21:34	1500
16370	06/12/05	12:08	2000	16418	06/12/05	21:38	3500
16371	06/12/05	12:16	200	16419	06/12/05	22:26	2000
16372	06/12/05	12:17	300	16420	07/12/05	00:17	1000
16373	06/12/05	12:18	12000	16421	07/12/05	06:44	200
16374	06/12/05	12:19	5000	16422	07/12/05	08:32	700
16375	06/12/05	12:37	1000	16423	07/12/05	09:38	7000
16376	06/12/05	13:42	3000	16424	07/12/05	09:42	200
16377	06/12/05	13:43	500	16425	07/12/05	09:43	100
16378	06/12/05	13:44	7800	16426	07/12/05	10:13	500
16379	06/12/05	13:47	5000	16427	07/12/05	10:15	5000
16380	06/12/05	14:24	2300	16428	07/12/05	10:31	2200
16381	06/12/05	14:27	15000	16429	07/12/05	10:56	800
16382	06/12/05	14:30	2300	16430	07/12/05	10:58	1200
16383	06/12/05	14:32	300	16431	07/12/05	11:05	1000
16384	06/12/05	15:12	2000	16432	07/12/05	11:08	100
16385	06/12/05	15:24	100	16433	07/12/05	11:53	500
16386	06/12/05	15:36	500	16434	07/12/05	11:57	2100
16387	06/12/05	15:42	1500	16435	07/12/05	12:06	1000
16388	06/12/05	15:54	1000	16436	07/12/05	12:40	700
16389	06/12/05	16:03	20000	16437	07/12/05	12:48	7000
16390	06/12/05	16:04	8000	16438	07/12/05	12:58	3000
16391	06/12/05	16:19	200	16439	07/12/05	13:04	100
16392	06/12/05	16:27	10500	16440	07/12/05	14:23	100
16393	06/12/05	16:27	200	16441	07/12/05	14:26	1000
16394	06/12/05	16:30	7000	16442	07/12/05	15:24	500
16395	06/12/05	16:34	10000	16443	07/12/05	15:26	100
16396	06/12/05	16:42	3200	16444	07/12/05	15:31	300
16397	06/12/05	16:45	3000	16445	07/12/05	15:33	1000
16398	06/12/05	17:00	500	16446	07/12/05	15:35	1300
16399	06/12/05	17:24	500	16447	07/12/05	15:39	20000
16400	06/12/05	17:37	2000	16448	07/12/05	15:40	5000
16401	06/12/05	17:53	700	16449	07/12/05	16:31	200
16402	06/12/05	18:02	13000	16450	07/12/05	16:47	1300
16403	06/12/05	18:36	200	16451	07/12/05	16:51	1000
16404	06/12/05	18:38	1000	16452	07/12/05	17:10	8000

16453	07/12/05	17:12	9000	16501	08/12/05	14:16	6800
16454	07/12/05	17:21	3000	16502	08/12/05	16:01	2200
16455	07/12/05	17:40	2000	16503	08/12/05	16:03	2300
16456	07/12/05	17:51	1200	16504	08/12/05	16:05	100
16457	07/12/05	17:54	200	16505	08/12/05	16:06	1000
16458	07/12/05	17:55	3000	16506	08/12/05	16:08	500
16459	07/12/05	17:56	3000	16507	08/12/05	16:24	8000
16460	07/12/05	18:10	2000	16508	08/12/05	17:00	900
16461	07/12/05	18:10	1000	16509	08/12/05	17:17	200
16462	07/12/05	18:11	1000	16510	08/12/05	17:22	1000
16463	07/12/05	18:14	100	16511	08/12/05	17:57	300
16464	07/12/05	18:21	700	16512	08/12/05	18:03	2000
16465	07/12/05	18:30	10000	16513	08/12/05	18:17	1000
16466	07/12/05	18:31	2500	16514	08/12/05	18:18	1000
16467	07/12/05	18:39	5000	16515	08/12/05	18:26	100
16468	07/12/05	18:55	2000	16516	08/12/05	19:22	2000
16469	07/12/05	19:23	3000	16517	08/12/05	19:33	10000
16470	07/12/05	20:33	100	16518	08/12/05	20:00	3500
16471	07/12/05	20:34	1000	16519	08/12/05	20:09	1000
16472	07/12/05	20:50	2000	16520	08/12/05	20:10	1000
16473	07/12/05	21:36	200	16521	08/12/05	20:21	20000
16474	07/12/05	21:39	100	16522	08/12/05	20:44	7000
16475	07/12/05	22:08	100	16523	08/12/05	23:14	8000
16476	07/12/05	22:27	2000	16524	09/12/05	00:07	1500
16477	07/12/05	22:27	3000	16525	09/12/05	00:23	100
16478	07/12/05	23:05	200	16526	09/12/05	09:21	700
16479	07/12/05	23:56	500	16527	09/12/05	09:34	3000
16480	08/12/05	08:32	2000	16528	09/12/05	09:35	2000
16481	08/12/05	09:19	3100	16529	09/12/05	10:00	2000
16482	08/12/05	09:21	2200	16530	09/12/05	10:35	100
16483	08/12/05	10:07	1200	16531	09/12/05	10:44	200
16484	08/12/05	10:09	2000	16532	09/12/05	11:29	500
16485	08/12/05	10:35	1500	16533	09/12/05	11:33	8000
16486	08/12/05	10:42	1200	16534	09/12/05	11:36	10000
16487	08/12/05	10:45	1100	16535	09/12/05	11:39	8000
16488	08/12/05	10:55	500	16536	09/12/05	11:47	1200
16489	08/12/05	11:12	200	16537	09/12/05	11:49	500
16490	08/12/05	11:18	2500	16538	09/12/05	12:09	2000
16491	08/12/05	11:28	1500	16539	09/12/05	12:10	20000
16492	08/12/05	11:29	100	16540	09/12/05	12:29	1000
16493	08/12/05	11:50	5500	16541	09/12/05	12:43	3000
16494	08/12/05	11:57	1300	16542	09/12/05	12:44	3000
16495	08/12/05	12:44	10000	16543	09/12/05	13:03	300
16496	08/12/05	12:57	500	16544	09/12/05	13:26	500
16497	08/12/05	13:32	500	16545	09/12/05	13:43	500
16498	08/12/05	13:47	200	16546	09/12/05	13:52	16500
16499	08/12/05	13:51	10000	16547	09/12/05	14:16	2000
16500	08/12/05	13:59	100	16548	09/12/05	14:24	500

16549	09/12/05	14:49	1500
16550	09/12/05	15:14	1000
16551	09/12/05	16:42	300
16552	09/12/05	17:32	1500
16553	09/12/05	18:00	500
16554	09/12/05	18:00	500
16555	09/12/05	18:01	100
16556	09/12/05	18:30	5000
16557	09/12/05	18:42	100
16558	09/12/05	18:50	200
16559	09/12/05	19:25	1500
16560	09/12/05	19:30	100
16561	09/12/05	19:33	200
16562	09/12/05	19:43	100
16563	09/12/05	20:01	800
16564	09/12/05	20:04	100
16565	09/12/05	20:07	500
16566	09/12/05	20:19	3000
16567	09/12/05	20:30	200
16568	09/12/05	20:32	1000
16569	09/12/05	22:28	2000
16570	10/12/05	07:44	2000
16571	10/12/05	07:45	500
16572	10/12/05	07:46	500
16573	10/12/05	09:48	2000
16574	10/12/05	10:01	500
16575	10/12/05	10:02	15000
16576	10/12/05	10:06	1000
16577	10/12/05	10:08	1000
16578	10/12/05	10:37	2000
16579	10/12/05	11:01	300
16580	10/12/05	11:03	5000
16581	10/12/05	11:08	1000
16582	10/12/05	11:18	2000
16583	10/12/05	11:19	300
16584	10/12/05	11:28	1000
16585	10/12/05	11:34	3700
16586	10/12/05	11:38	8000
16587	10/12/05	11:43	200
16588	10/12/05	11:47	9200
16589	10/12/05	11:54	2000
16590	10/12/05	11:55	200
16591	10/12/05	11:56	500
16592	10/12/05	12:13	1000
16593	10/12/05	12:25	300
16594	10/12/05	12:27	500
16595	10/12/05	12:33	1000
16596	10/12/05	12:38	3000
16597	10/12/05	13:06	1100
16598	10/12/05	13:27	2300
16599	10/12/05	13:37	300
16600	10/12/05	13:42	100
16601	10/12/05	13:51	700
16602	10/12/05	13:59	2000
16603	10/12/05	14:13	17000
16604	10/12/05	14:41	1000
16605	10/12/05	14:59	2000
16606	10/12/05	15:33	8000
16607	10/12/05	15:34	8000
16608	10/12/05	15:52	15000
16609	10/12/05	15:58	300
16610	10/12/05	16:24	1000
16611	10/12/05	16:35	12000
16612	10/12/05	16:41	5000
16613	10/12/05	16:55	2000
16614	10/12/05	16:58	2000
16615	10/12/05	17:26	500
16616	10/12/05	17:50	200
16617	10/12/05	17:51	500
16618	10/12/05	17:57	100
16619	10/12/05	18:06	1100
16620	10/12/05	18:07	200
16621	10/12/05	18:08	100
16622	10/12/05	18:10	3000
16623	10/12/05	18:10	2000
16624	10/12/05	18:13	1000
16625	10/12/05	18:25	200
16626	10/12/05	18:34	500
16627	10/12/05	18:38	200
16628	10/12/05	18:47	500
16629	10/12/05	18:56	1300
16630	10/12/05	19:05	1800
16631	10/12/05	19:15	500
16632	10/12/05	19:35	1000
16633	10/12/05	19:37	3000
16634	10/12/05	19:45	300
16635	10/12/05	20:33	800
16636	10/12/05	20:50	100
16637	10/12/05	21:14	100
16638	10/12/05	21:23	3100
16639	10/12/05	21:29	2000
16640	10/12/05	21:31	500
16641	10/12/05	21:40	900
16642	10/12/05	22:08	200
16643	11/12/05	00:25	300
16644	11/12/05	01:14	100

16645	11/12/05	09:42	20000
16646	11/12/05	09:43	8000
16647	11/12/05	10:30	1000
16648	11/12/05	10:50	3000
16649	11/12/05	11:11	300
16650	11/12/05	11:24	2000
16651	11/12/05	11:51	1500
16652	11/12/05	11:58	2000
16653	11/12/05	12:06	1100
16654	11/12/05	12:11	10200
16655	11/12/05	12:31	20000
16656	11/12/05	12:33	15000
16657	11/12/05	12:46	2000
16658	11/12/05	13:14	2000
16659	11/12/05	13:22	1500
16660	11/12/05	13:26	2000
16661	11/12/05	14:18	1200
16662	11/12/05	15:09	2000
16663	11/12/05	15:39	500
16664	11/12/05	15:41	100
16665	11/12/05	16:40	500
16666	11/12/05	16:54	5000
16667	11/12/05	17:04	500
16668	11/12/05	17:05	500
16669	11/12/05	17:15	500
16670	11/12/05	17:28	200
16671	11/12/05	17:40	10000
16672	11/12/05	17:47	1000
16673	11/12/05	17:52	500
16674	11/12/05	18:10	5000
16675	11/12/05	18:14	300
16676	11/12/05	18:23	1000
16677	11/12/05	18:24	1000
16678	11/12/05	18:25	1000
16679	11/12/05	18:35	1000
16680	11/12/05	18:38	200
16681	11/12/05	18:59	3500
16682	11/12/05	19:07	500
16683	11/12/05	19:22	2500
16684	11/12/05	19:27	20000
16685	11/12/05	19:38	1000
16686	11/12/05	19:43	100
16687	11/12/05	19:55	900
16688	11/12/05	20:10	500
16689	11/12/05	20:16	1000
16690	11/12/05	21:24	3000
16691	11/12/05	22:13	5000
16692	11/12/05	22:27	500
16693	11/12/05	22:32	3000
16694	11/12/05	23:06	200
16695	11/12/05	23:07	1500
16696	11/12/05	23:29	300
16697	11/12/05	23:35	300
16698	11/12/05	23:45	1000
16699	12/12/05	08:58	17000
16700	12/12/05	09:49	500
16701	12/12/05	11:04	10000
16702	12/12/05	11:37	1200
16703	12/12/05	11:58	1000
16704	12/12/05	12:16	2000
16705	12/12/05	12:29	500
16706	12/12/05	12:32	3000
16707	12/12/05	12:45	4500
16708	12/12/05	12:47	5000
16709	12/12/05	13:04	10000
16710	12/12/05	13:34	22000
16711	12/12/05	13:40	100
16712	12/12/05	14:21	3000
16713	12/12/05	15:00	500
16714	12/12/05	15:07	1000
16715	12/12/05	16:01	1000
16716	12/12/05	16:14	1200
16717	12/12/05	16:50	9000
16718	12/12/05	17:06	3000
16719	12/12/05	17:08	200
16720	12/12/05	17:21	500
16721	12/12/05	18:10	100
16722	12/12/05	18:19	13000
16723	12/12/05	18:59	3000
16724	12/12/05	19:17	200
16725	12/12/05	19:24	300
16726	12/12/05	19:37	2000
16727	12/12/05	20:30	3000
16728	12/12/05	20:55	15000
16729	12/12/05	21:06	500
16730	12/12/05	21:20	200
16731	12/12/05	23:49	100
16732	12/12/05	23:51	100
16733	12/12/05	23:52	100
16734	13/12/05	01:31	100
16735	13/12/05	09:11	200
16736	13/12/05	09:35	3000
16737	13/12/05	09:36	3000
16738	13/12/05	10:35	1000
16739	13/12/05	10:36	1000
16740	13/12/05	10:48	500

16741	13/12/05	10:49	300
16742	13/12/05	10:56	1000
16743	13/12/05	11:00	2000
16744	13/12/05	11:23	8000
16745	13/12/05	11:25	15000
16746	13/12/05	11:29	10000
16747	13/12/05	11:32	5000
16748	13/12/05	11:38	100
16749	13/12/05	11:49	11000
16750	13/12/05	11:52	5000
16751	13/12/05	11:53	1100
16752	13/12/05	12:48	5000
16753	13/12/05	12:51	2000
16754	13/12/05	13:03	2100
16755	13/12/05	13:18	10000
16756	13/12/05	13:30	200
16757	13/12/05	14:13	2000
16758	13/12/05	14:17	20000
16759	13/12/05	14:33	3000
16760	13/12/05	14:36	15000
16761	13/12/05	15:18	1500
16762	13/12/05	15:29	3500
16763	13/12/05	15:47	13000
16764	13/12/05	15:49	1000
16765	13/12/05	16:02	1200
16766	13/12/05	16:17	1000
16767	13/12/05	16:47	23000
16768	13/12/05	16:54	1500
16769	13/12/05	16:58	3000
16770	13/12/05	17:01	11500
16771	13/12/05	17:11	2000
16772	13/12/05	17:27	200
16773	13/12/05	17:29	100
16774	13/12/05	18:10	100
16775	13/12/05	18:13	10000
16776	13/12/05	18:18	12000
16777	13/12/05	18:21	300
16778	13/12/05	18:24	10000
16779	13/12/05	18:33	2000
16780	13/12/05	18:37	3500
16781	13/12/05	18:56	100
16782	13/12/05	19:08	3000
16783	13/12/05	19:11	1000
16784	13/12/05	19:32	1700
16785	13/12/05	19:36	500
16786	13/12/05	20:39	3200
16787	13/12/05	20:51	500
16788	13/12/05	20:59	500
16789	13/12/05	21:06	20000
16790	13/12/05	21:08	8000
16791	13/12/05	21:11	8000
16792	13/12/05	21:20	100
16793	13/12/05	21:37	1200
16794	13/12/05	22:27	500
16795	14/12/05	08:53	1500
16796	14/12/05	09:06	1000
16797	14/12/05	09:08	3000
16798	14/12/05	10:16	12000
16799	14/12/05	11:00	5500
16800	14/12/05	11:02	5500
16801	14/12/05	11:04	1000
16802	14/12/05	11:06	5000
16803	14/12/05	11:29	100
16804	14/12/05	11:37	10000
16805	14/12/05	11:49	500
16806	14/12/05	11:49	200
16807	14/12/05	11:56	1000
16808	14/12/05	12:22	3000
16809	14/12/05	12:29	500
16810	14/12/05	12:50	700
16811	14/12/05	12:57	800
16812	14/12/05	13:19	1000
16813	14/12/05	13:26	2000
16814	14/12/05	13:27	3000
16815	14/12/05	13:34	22000
16816	14/12/05	13:37	20000
16817	14/12/05	13:40	8000
16818	14/12/05	13:41	2000
16819	14/12/05	13:43	20000
16820	14/12/05	13:47	15000
16821	14/12/05	14:02	8000
16822	14/12/05	14:37	3000
16823	14/12/05	15:18	900
16824	14/12/05	15:52	1000
16825	14/12/05	16:01	3000
16826	14/12/05	16:08	15000
16827	14/12/05	16:11	10000
16828	14/12/05	16:18	200
16829	14/12/05	16:43	2000
16830	14/12/05	17:05	1500
16831	14/12/05	17:26	700
16832	14/12/05	17:40	1500
16833	14/12/05	18:17	1200
16834	14/12/05	18:27	1000
16835	14/12/05	18:28	9000
16836	14/12/05	18:29	500

16837	14/12/05	18:33	300
16838	14/12/05	19:04	20000
16839	14/12/05	19:06	8000
16840	14/12/05	19:08	1000
16841	14/12/05	19:34	300
16842	14/12/05	19:56	200
16843	14/12/05	20:46	5000
16844	15/12/05	07:42	300
16845	15/12/05	09:41	500
16846	15/12/05	09:48	20000
16847	15/12/05	09:49	5000
16848	15/12/05	09:51	500
16849	15/12/05	10:14	200
16850	15/12/05	10:26	100
16851	15/12/05	10:40	100
16852	15/12/05	10:41	1000
16853	15/12/05	10:47	700
16854	15/12/05	10:48	500
16855	15/12/05	10:55	19700
16856	15/12/05	10:57	200
16857	15/12/05	11:10	3000
16858	15/12/05	11:12	500
16859	15/12/05	11:45	200
16860	15/12/05	12:00	4500
16861	15/12/05	12:11	500
16862	15/12/05	12:20	20000
16863	15/12/05	12:22	8000
16864	15/12/05	12:51	1000
16865	15/12/05	12:53	5000
16866	15/12/05	13:16	700
16867	15/12/05	13:20	300
16868	15/12/05	13:21	500
16869	15/12/05	14:04	20000
16870	15/12/05	14:06	12000
16871	15/12/05	14:26	3200
16872	15/12/05	14:27	8000
16873	15/12/05	14:28	2000
16874	15/12/05	16:01	1000
16875	15/12/05	16:20	7000
16876	15/12/05	16:35	5000
16877	15/12/05	16:58	5000
16878	15/12/05	16:59	5000
16879	15/12/05	17:56	200
16880	15/12/05	18:00	5000
16881	15/12/05	18:10	2000
16882	15/12/05	18:12	2500
16883	15/12/05	18:24	10000
16884	15/12/05	18:53	800
16885	15/12/05	19:00	21000
16886	15/12/05	19:01	1000
16887	15/12/05	19:11	100
16888	15/12/05	19:12	1000
16889	15/12/05	19:14	100
16890	15/12/05	19:21	500
16891	15/12/05	19:24	500
16892	15/12/05	21:11	200
16893	15/12/05	21:27	16500
16894	15/12/05	21:32	500
16895	15/12/05	22:38	3000
16896	15/12/05	22:43	1000
16897	16/12/05	00:32	20000
16898	16/12/05	06:13	500
16899	16/12/05	06:13	5000
16900	16/12/05	09:24	2500
16901	16/12/05	09:41	1000
16902	16/12/05	09:50	200
16903	16/12/05	09:52	20000
16904	16/12/05	09:52	8000
16905	16/12/05	09:54	2000
16906	16/12/05	11:06	5000
16907	16/12/05	11:28	1000
16908	16/12/05	12:00	100
16909	16/12/05	12:28	1000
16910	16/12/05	12:36	300
16911	16/12/05	12:55	7000
16912	16/12/05	13:08	500
16913	16/12/05	13:09	3000
16914	16/12/05	14:40	300
16915	16/12/05	17:11	12000
16916	16/12/05	17:16	200
16917	16/12/05	17:22	1000
16918	16/12/05	18:10	20000
16919	16/12/05	19:19	3000
16920	16/12/05	19:27	2000
16921	16/12/05	19:43	1000
16922	16/12/05	20:29	200
16923	16/12/05	22:56	200
16924	17/12/05	09:39	2000
16925	17/12/05	09:53	500
16926	17/12/05	10:00	5000
16927	17/12/05	10:02	500
16928	17/12/05	10:03	3000
16929	17/12/05	10:10	10000
16930	17/12/05	10:29	3300
16931	17/12/05	10:33	200
16932	17/12/05	10:39	1800

16933	17/12/05	11:09	1000
16934	17/12/05	11:51	1700
16935	17/12/05	11:54	17000
16936	17/12/05	11:56	100
16937	17/12/05	12:03	20000
16938	17/12/05	12:04	8000
16939	17/12/05	12:05	8000
16940	17/12/05	12:27	1000
16941	17/12/05	12:45	500
16942	17/12/05	13:27	1000
16943	17/12/05	14:29	2000
16944	17/12/05	15:06	2000
16945	17/12/05	15:19	20000
16946	17/12/05	15:20	10000
16947	17/12/05	15:21	20000
16948	17/12/05	15:52	5000
16949	17/12/05	15:52	10000
16950	17/12/05	15:54	1000
16951	17/12/05	15:55	1200
16952	17/12/05	15:59	2000
16953	17/12/05	16:25	5000
16954	17/12/05	16:52	5500
16955	17/12/05	17:33	700
16956	17/12/05	17:34	100
16957	17/12/05	17:37	17000
16958	17/12/05	17:49	3000
16959	17/12/05	18:11	200
16960	17/12/05	18:12	200
16961	17/12/05	18:12	200
16962	17/12/05	18:13	200
16963	17/12/05	18:14	200
16964	17/12/05	18:16	500
16965	17/12/05	18:24	12000
16966	17/12/05	18:26	1000
16967	17/12/05	18:28	4500
16968	17/12/05	18:37	500
16969	17/12/05	19:15	3000
16970	17/12/05	19:17	3000
16971	17/12/05	19:19	1500
16972	17/12/05	19:30	3000
16973	17/12/05	19:31	2000
16974	17/12/05	19:57	2000
16975	17/12/05	20:06	500
16976	17/12/05	22:09	2000
16977	17/12/05	22:34	100
16978	18/12/05	11:08	3500
16979	18/12/05	11:27	200
16980	18/12/05	11:36	4800
16981	18/12/05	11:49	1500
16982	18/12/05	11:56	200
16983	18/12/05	12:20	1000
16984	18/12/05	12:21	20000
16985	18/12/05	12:22	8000
16986	18/12/05	12:28	8000
16987	18/12/05	12:29	100
16988	18/12/05	12:35	5000
16989	18/12/05	12:57	500
16990	18/12/05	13:28	100
16991	18/12/05	13:53	2000
16992	18/12/05	14:27	200
16993	18/12/05	14:34	100
16994	18/12/05	14:42	300
16995	18/12/05	15:05	2000
16996	18/12/05	15:10	2000
16997	18/12/05	15:52	2000
16998	18/12/05	16:30	3000
16999	18/12/05	16:45	1200
17000	18/12/05	18:20	20000
17001	18/12/05	18:59	300
17002	18/12/05	19:04	1000
17003	18/12/05	19:05	1000
17004	18/12/05	19:44	2000
17005	18/12/05	20:36	300
17006	18/12/05	20:50	500
17007	18/12/05	21:10	10000
17008	18/12/05	21:15	200
17009	18/12/05	21:24	500
17010	18/12/05	21:25	1000
17011	18/12/05	21:26	500
17012	18/12/05	21:27	3000
17013	18/12/05	22:23	3000
17014	18/12/05	23:01	20000
17015	18/12/05	23:02	8000
17016	19/12/05	07:15	15000
17017	19/12/05	09:14	300
17018	19/12/05	09:40	500
17019	19/12/05	09:56	1500
17020	19/12/05	10:16	1500
17021	19/12/05	10:32	20000
17022	19/12/05	10:36	3000
17023	19/12/05	10:38	10000
17024	19/12/05	11:00	100
17025	19/12/05	11:07	700
17026	19/12/05	11:22	800
17027	19/12/05	11:53	5000
17028	19/12/05	11:57	20000

17029	19/12/05	11:58	8000
17030	19/12/05	12:15	1200
17031	19/12/05	12:17	1000
17032	19/12/05	12:19	3500
17033	19/12/05	12:27	10000
17034	19/12/05	12:28	1200
17035	19/12/05	12:30	100
17036	19/12/05	12:32	1000
17037	19/12/05	12:57	500
17038	19/12/05	13:23	500
17039	19/12/05	13:28	500
17040	19/12/05	14:46	20000
17041	19/12/05	14:47	12000
17042	19/12/05	14:50	2000
17043	19/12/05	14:52	3000
17044	19/12/05	16:29	100
17045	19/12/05	16:33	3300
17046	19/12/05	16:49	3000
17047	19/12/05	17:08	2000
17048	19/12/05	17:15	500
17049	19/12/05	17:16	200
17050	19/12/05	17:38	4300
17051	19/12/05	17:42	500
17052	19/12/05	17:43	1000
17053	19/12/05	18:12	200
17054	19/12/05	18:17	20000
17055	19/12/05	18:26	3000
17056	19/12/05	18:30	500
17057	19/12/05	18:37	2000
17058	19/12/05	18:39	500
17059	19/12/05	19:01	3000
17060	19/12/05	19:02	3000
17061	19/12/05	19:03	500
17062	19/12/05	19:09	6900
17063	19/12/05	19:17	5000
17064	19/12/05	19:24	200
17065	19/12/05	19:46	2500
17066	19/12/05	19:57	1500
17067	19/12/05	20:04	1000
17068	19/12/05	20:16	2000
17069	19/12/05	21:47	1000
17070	19/12/05	21:53	2000
17071	19/12/05	21:56	500
17072	19/12/05	23:20	1000
17073	19/12/05	23:34	100
17074	20/12/05	08:42	8000
17075	20/12/05	08:49	100
17076	20/12/05	08:50	8000
17077	20/12/05	10:20	20000
17078	20/12/05	11:08	1000
17079	20/12/05	11:09	2000
17080	20/12/05	11:28	700
17081	20/12/05	11:36	3000
17082	20/12/05	12:21	3500
17083	20/12/05	12:23	500
17084	20/12/05	12:24	1000
17085	20/12/05	12:36	1200
17086	20/12/05	13:23	500
17087	20/12/05	13:31	10000
17088	20/12/05	13:41	2000
17089	20/12/05	14:15	5000
17090	20/12/05	14:18	3000
17091	20/12/05	14:38	8000
17092	20/12/05	14:39	20000
17093	20/12/05	14:50	200
17094	20/12/05	15:18	2500
17095	20/12/05	16:31	1500
17096	20/12/05	17:01	100
17097	20/12/05	17:19	7000
17098	20/12/05	18:06	1500
17099	20/12/05	18:07	200
17100	20/12/05	18:09	8000
17101	20/12/05	18:29	6100
17102	20/12/05	18:31	7500
17103	20/12/05	19:45	300
17104	20/12/05	19:49	200
17105	20/12/05	20:11	20000
17106	20/12/05	20:12	7000
17107	20/12/05	20:13	1000
17108	20/12/05	20:14	300
17109	20/12/05	20:15	1000
17110	20/12/05	20:43	1000
17111	20/12/05	20:49	500
17112	20/12/05	20:50	500
17113	20/12/05	20:51	1000
17114	20/12/05	20:53	200
17115	20/12/05	21:33	100
17116	20/12/05	21:34	300
17117	20/12/05	21:43	1000
17118	20/12/05	21:45	100
17119	20/12/05	22:16	300
17120	20/12/05	22:22	5100
17121	21/12/05	09:16	2000
17122	21/12/05	09:17	500
17123	21/12/05	09:47	200
17124	21/12/05	11:13	300

17125	21/12/05	11:17	8000
17126	21/12/05	11:41	200
17127	21/12/05	11:44	100
17128	21/12/05	11:53	5000
17129	21/12/05	12:13	500
17130	21/12/05	12:21	100
17131	21/12/05	12:38	6500
17132	21/12/05	12:39	100
17133	21/12/05	12:42	2000
17134	21/12/05	13:06	8000
17135	21/12/05	13:06	8000
17136	21/12/05	13:09	500
17137	21/12/05	13:14	100
17138	21/12/05	13:18	1000
17139	21/12/05	13:25	10000
17140	21/12/05	13:29	100
17141	21/12/05	13:32	13000
17142	21/12/05	13:34	15000
17143	21/12/05	14:09	3000
17144	21/12/05	14:31	1500
17145	21/12/05	14:33	2000
17146	21/12/05	14:37	200
17147	21/12/05	14:52	500
17148	21/12/05	15:09	100
17149	21/12/05	15:27	3000
17150	21/12/05	15:30	20000
17151	21/12/05	16:08	500
17152	21/12/05	16:11	200
17153	21/12/05	16:17	2500
17154	21/12/05	16:54	5000
17155	21/12/05	17:17	300
17156	21/12/05	17:18	1000
17157	21/12/05	17:32	5000
17158	21/12/05	17:56	700
17159	21/12/05	18:17	1800
17160	21/12/05	18:20	1000
17161	21/12/05	18:25	1000
17162	21/12/05	18:45	2000
17163	21/12/05	18:58	1200
17164	21/12/05	19:22	10000
17165	21/12/05	19:24	2000
17166	21/12/05	19:27	1200
17167	21/12/05	19:36	1000
17168	21/12/05	19:41	7000
17169	21/12/05	19:49	100
17170	21/12/05	19:56	100
17171	21/12/05	19:59	3000
17172	21/12/05	20:35	1800
17173	21/12/05	20:37	700
17174	21/12/05	22:23	500
17175	22/12/05	05:46	100
17176	22/12/05	06:09	500
17177	22/12/05	06:39	3500
17178	22/12/05	08:01	2000
17179	22/12/05	08:05	500
17180	22/12/05	09:00	18000
17181	22/12/05	09:17	1000
17182	22/12/05	10:03	800
17183	22/12/05	10:25	500
17184	22/12/05	10:55	17800
17185	22/12/05	10:59	7500
17186	22/12/05	11:02	100
17187	22/12/05	11:04	100
17188	22/12/05	11:08	300
17189	22/12/05	11:10	3900
17190	22/12/05	11:35	8000
17191	22/12/05	11:36	8000
17192	22/12/05	11:37	3000
17193	22/12/05	11:42	200
17194	22/12/05	11:51	1000
17195	22/12/05	11:52	500
17196	22/12/05	12:05	100
17197	22/12/05	12:05	1500
17198	22/12/05	12:14	10000
17199	22/12/05	12:18	500
17200	22/12/05	12:27	2000
17201	22/12/05	12:49	1000
17202	22/12/05	12:51	200
17203	22/12/05	12:53	2000
17204	22/12/05	12:58	1000
17205	22/12/05	12:59	8000
17206	22/12/05	13:03	4500
17207	22/12/05	13:06	2000
17208	22/12/05	13:10	8700
17209	22/12/05	13:11	300
17210	22/12/05	13:18	100
17211	22/12/05	13:30	7000
17212	22/12/05	14:09	20000
17213	22/12/05	14:11	8000
17214	22/12/05	14:21	1000
17215	22/12/05	14:42	3000
17216	22/12/05	14:43	10000
17217	22/12/05	14:56	1000
17218	22/12/05	15:07	1000
17219	22/12/05	17:31	500
17220	22/12/05	17:37	200

17221	22/12/05	17:50	10000
17222	22/12/05	18:04	2000
17223	22/12/05	18:05	200
17224	22/12/05	18:29	1000
17225	22/12/05	19:03	1000
17226	22/12/05	19:15	2000
17227	22/12/05	19:50	5300
17228	22/12/05	19:52	500
17229	22/12/05	19:54	2000
17230	22/12/05	20:04	300
17231	22/12/05	21:19	100
17232	22/12/05	22:11	1000
17233	22/12/05	22:23	500
17234	23/12/05	07:52	1000
17235	23/12/05	08:01	500
17236	23/12/05	08:49	500
17237	23/12/05	08:57	1500
17238	23/12/05	09:30	500
17239	23/12/05	09:36	3000
17240	23/12/05	09:56	500
17241	23/12/05	10:11	1000
17242	23/12/05	10:20	3000
17243	23/12/05	10:23	2000
17244	23/12/05	10:28	2000
17245	23/12/05	10:33	5000
17246	23/12/05	11:23	2000
17247	23/12/05	11:41	500
17248	23/12/05	11:48	900
17249	23/12/05	11:58	1000
17250	23/12/05	12:00	900
17251	23/12/05	12:19	500
17252	23/12/05	12:28	300
17253	23/12/05	12:30	1000
17254	23/12/05	12:46	2000
17255	23/12/05	12:58	200
17256	23/12/05	13:38	5000
17257	23/12/05	13:44	100
17258	23/12/05	13:59	2500
17259	23/12/05	15:07	20000
17260	23/12/05	16:02	10000
17261	23/12/05	17:05	3000
17262	23/12/05	18:21	500
17263	23/12/05	18:29	500
17264	23/12/05	18:40	5000
17265	23/12/05	18:44	1500
17266	23/12/05	18:47	100
17267	23/12/05	19:51	5000
17268	23/12/05	20:07	2000
17269	23/12/05	20:28	1000
17270	23/12/05	21:06	100
17271	23/12/05	21:16	100
17272	23/12/05	21:40	10000
17273	23/12/05	22:01	500
17274	24/12/05	09:13	100
17275	24/12/05	09:52	300
17276	24/12/05	10:12	15000
17277	24/12/05	10:18	100
17278	24/12/05	10:19	2500
17279	24/12/05	10:21	2000
17280	24/12/05	10:22	2100
17281	24/12/05	10:25	200
17282	24/12/05	10:27	5500
17283	24/12/05	10:40	500
17284	24/12/05	10:53	100
17285	24/12/05	10:55	500
17286	24/12/05	11:17	1000
17287	24/12/05	11:27	2500
17288	24/12/05	11:31	100
17289	24/12/05	11:35	3000
17290	24/12/05	11:41	15000
17291	24/12/05	11:43	2000
17292	24/12/05	11:53	1000
17293	24/12/05	12:09	2000
17294	24/12/05	12:26	500
17295	24/12/05	12:28	20000
17296	24/12/05	12:38	500
17297	24/12/05	12:39	1500
17298	24/12/05	12:54	300
17299	24/12/05	13:00	500
17300	24/12/05	13:25	500
17301	24/12/05	13:33	10000
17302	24/12/05	13:47	5000
17303	24/12/05	13:52	700
17304	24/12/05	13:57	1000
17305	24/12/05	13:58	500
17306	24/12/05	13:58	500
17307	24/12/05	14:09	1000
17308	24/12/05	14:22	8000
17309	24/12/05	14:33	20000
17310	24/12/05	14:44	1000
17311	24/12/05	14:45	5000
17312	24/12/05	14:54	2000
17313	24/12/05	15:10	10000
17314	24/12/05	15:12	2000
17315	24/12/05	15:19	2000
17316	24/12/05	15:20	5000

17317	24/12/05	15:56	500
17318	24/12/05	16:27	15000
17319	24/12/05	17:23	12500
17320	24/12/05	17:29	2500
17321	24/12/05	17:34	6500
17322	24/12/05	17:35	2000
17323	24/12/05	18:08	10000
17324	24/12/05	18:10	3000
17325	24/12/05	18:11	3000
17326	24/12/05	18:14	5000
17327	24/12/05	18:37	1500
17328	24/12/05	18:39	1000
17329	24/12/05	18:49	500
17330	24/12/05	23:24	1000
17331	25/12/05	05:27	2200
17332	25/12/05	07:10	2000
17333	25/12/05	08:02	100
17334	25/12/05	08:06	500
17335	25/12/05	09:26	1100
17336	25/12/05	09:29	1000
17337	25/12/05	09:49	1000
17338	25/12/05	10:35	2000
17339	25/12/05	10:56	8000
17340	25/12/05	11:07	1500
17341	25/12/05	11:12	100
17342	25/12/05	11:22	500
17343	25/12/05	11:30	20000
17344	25/12/05	11:33	500
17345	25/12/05	11:37	300
17346	25/12/05	11:50	3000
17347	25/12/05	12:19	500
17348	25/12/05	12:31	1000
17349	25/12/05	12:45	500
17350	25/12/05	13:12	2500
17351	25/12/05	13:15	3000
17352	25/12/05	13:16	3500
17353	25/12/05	13:20	2000
17354	25/12/05	13:27	15000
17355	25/12/05	13:40	1500
17356	25/12/05	14:21	200
17357	25/12/05	14:42	9000
17358	25/12/05	16:00	20000
17359	25/12/05	16:13	500
17360	25/12/05	16:51	500
17361	25/12/05	16:53	2000
17362	25/12/05	17:32	3000
17363	25/12/05	17:47	8000
17364	25/12/05	17:48	8000
17365	25/12/05	18:01	1200
17366	25/12/05	18:26	3000
17367	25/12/05	18:30	500
17368	25/12/05	18:43	900
17369	25/12/05	18:44	200
17370	25/12/05	18:54	300
17371	25/12/05	19:55	5000
17372	25/12/05	20:00	200
17373	25/12/05	22:56	200
17374	26/12/05	00:43	1500
17375	26/12/05	00:57	18700
17376	26/12/05	07:44	100
17377	26/12/05	08:10	2000
17378	26/12/05	08:49	6500
17379	26/12/05	09:04	1000
17380	26/12/05	09:16	20000
17381	26/12/05	10:05	1000
17382	26/12/05	10:15	200
17383	26/12/05	10:53	500
17384	26/12/05	10:55	2000
17385	26/12/05	10:56	1000
17386	26/12/05	11:02	500
17387	26/12/05	11:45	1000
17388	26/12/05	11:57	200
17389	26/12/05	12:26	1000
17390	26/12/05	12:31	8000
17391	26/12/05	12:32	8000
17392	26/12/05	12:49	2000
17393	26/12/05	12:54	100
17394	26/12/05	12:58	2500
17395	26/12/05	13:02	3200
17396	26/12/05	13:24	500
17397	26/12/05	13:33	5000
17398	26/12/05	14:04	5000
17399	26/12/05	14:05	200
17400	26/12/05	14:33	5000
17401	26/12/05	14:51	20000
17402	26/12/05	14:51	5000
17403	26/12/05	15:20	2000
17404	26/12/05	16:46	2000
17405	26/12/05	16:47	800
17406	26/12/05	17:03	500
17407	26/12/05	17:11	2200
17408	26/12/05	17:16	3000
17409	26/12/05	17:26	5000
17410	26/12/05	17:27	3000
17411	26/12/05	17:27	1000
17412	26/12/05	17:28	500

17413	26/12/05	17:39	7000
17414	26/12/05	17:54	2000
17415	26/12/05	18:03	2000
17416	26/12/05	18:39	100
17417	26/12/05	18:40	100
17418	26/12/05	18:48	3000
17419	26/12/05	18:53	2000
17420	26/12/05	18:54	1000
17421	26/12/05	19:29	10000
17422	26/12/05	19:32	3000
17423	26/12/05	19:42	1700
17424	26/12/05	19:44	100
17425	26/12/05	19:49	200
17426	26/12/05	20:16	500
17427	26/12/05	20:44	100
17428	26/12/05	20:48	200
17429	26/12/05	21:05	1000
17430	26/12/05	21:48	2000
17431	26/12/05	22:47	20000
17432	27/12/05	08:12	100
17433	27/12/05	09:11	10000
17434	27/12/05	09:14	1000
17435	27/12/05	09:50	6300
17436	27/12/05	10:19	1200
17437	27/12/05	10:46	1000
17438	27/12/05	11:04	7000
17439	27/12/05	11:05	7000
17440	27/12/05	11:06	100
17441	27/12/05	11:21	8000
17442	27/12/05	11:24	1300
17443	27/12/05	11:26	200
17444	27/12/05	11:29	100
17445	27/12/05	11:34	17500
17446	27/12/05	11:41	8000
17447	27/12/05	11:43	20000
17448	27/12/05	12:02	5500
17449	27/12/05	12:13	15000
17450	27/12/05	12:16	10000
17451	27/12/05	12:21	2000
17452	27/12/05	12:22	10000
17453	27/12/05	12:24	2000
17454	27/12/05	12:26	6500
17455	27/12/05	12:33	5000
17456	27/12/05	12:35	10000
17457	27/12/05	12:36	2000
17458	27/12/05	12:37	2000
17459	27/12/05	12:52	2000
17460	27/12/05	12:52	1000
17461	27/12/05	13:07	500
17462	27/12/05	13:12	2000
17463	27/12/05	13:13	200
17464	27/12/05	13:20	300
17465	27/12/05	13:23	15000
17466	27/12/05	13:27	500
17467	27/12/05	13:28	5000
17468	27/12/05	13:39	10500
17469	27/12/05	13:41	15000
17470	27/12/05	13:53	8000
17471	27/12/05	13:54	20000
17472	27/12/05	14:17	1000
17473	27/12/05	14:18	5000
17474	27/12/05	14:21	100
17475	27/12/05	14:43	300
17476	27/12/05	14:45	3000
17477	27/12/05	14:47	5000
17478	27/12/05	14:51	2000
17479	27/12/05	14:57	100
17480	27/12/05	15:03	2300
17481	27/12/05	15:17	1200
17482	27/12/05	15:27	300
17483	27/12/05	15:37	5000
17484	27/12/05	15:44	500
17485	27/12/05	15:58	5000
17486	27/12/05	16:06	2000
17487	27/12/05	16:12	1000
17488	27/12/05	16:22	1000
17489	27/12/05	17:09	2000
17490	27/12/05	17:10	15000
17491	27/12/05	17:33	10000
17492	27/12/05	17:54	500
17493	27/12/05	17:58	5900
17494	27/12/05	18:16	3500
17495	27/12/05	18:34	700
17496	27/12/05	18:41	200
17497	27/12/05	18:46	15000
17498	27/12/05	19:14	20000
17499	27/12/05	19:15	8000
17500	27/12/05	19:20	3000
17501	27/12/05	19:24	100
17502	27/12/05	19:26	2000
17503	27/12/05	19:28	2000
17504	27/12/05	19:30	3100
17505	27/12/05	19:31	2100
17506	27/12/05	19:36	2200
17507	27/12/05	20:00	300
17508	27/12/05	20:01	3300

17509	27/12/05	20:03	2000
17510	27/12/05	20:08	500
17511	27/12/05	20:13	500
17512	27/12/05	20:24	2000
17513	27/12/05	20:45	3700
17514	27/12/05	21:06	200
17515	27/12/05	23:26	1500
17516	28/12/05	08:12	2000
17517	28/12/05	08:13	300
17518	28/12/05	08:19	200
17519	28/12/05	08:29	20000
17520	28/12/05	09:23	3000
17521	28/12/05	09:24	3000
17522	28/12/05	09:25	3000
17523	28/12/05	09:26	1000
17524	28/12/05	09:29	3800
17525	28/12/05	09:30	2000
17526	28/12/05	09:34	5500
17527	28/12/05	10:22	3000
17528	28/12/05	10:23	20000
17529	28/12/05	10:23	5000
17530	28/12/05	10:25	7000
17531	28/12/05	10:26	7700
17532	28/12/05	10:42	300
17533	28/12/05	10:43	2500
17534	28/12/05	10:51	300
17535	28/12/05	10:53	20000
17536	28/12/05	10:55	2000
17537	28/12/05	10:58	1000
17538	28/12/05	11:00	1500
17539	28/12/05	11:08	300
17540	28/12/05	11:10	15000
17541	28/12/05	11:13	2000
17542	28/12/05	11:19	1000
17543	28/12/05	11:21	1000
17544	28/12/05	11:24	1500
17545	28/12/05	11:26	2000
17546	28/12/05	11:28	1000
17547	28/12/05	11:31	3500
17548	28/12/05	11:33	5000
17549	28/12/05	11:34	5000
17550	28/12/05	11:35	5000
17551	28/12/05	11:36	1500
17552	28/12/05	11:44	5000
17553	28/12/05	11:45	2000
17554	28/12/05	12:11	500
17555	28/12/05	12:15	2000
17556	28/12/05	12:25	500
17557	28/12/05	12:58	3000
17558	28/12/05	13:00	200
17559	28/12/05	13:05	2000
17560	28/12/05	13:07	2500
17561	28/12/05	13:08	500
17562	28/12/05	13:15	3000
17563	28/12/05	13:42	100
17564	28/12/05	14:37	2000
17565	28/12/05	14:38	2000
17566	28/12/05	14:40	300
17567	28/12/05	14:43	6300
17568	28/12/05	14:51	2000
17569	28/12/05	14:56	11000
17570	28/12/05	15:51	7000
17571	28/12/05	16:01	500
17572	28/12/05	16:02	1000
17573	28/12/05	16:07	1000
17574	28/12/05	16:15	10000
17575	28/12/05	16:20	3000
17576	28/12/05	16:25	2000
17577	28/12/05	16:26	20000
17578	28/12/05	16:27	8000
17579	28/12/05	16:30	3300
17580	28/12/05	16:38	500
17581	28/12/05	16:42	1000
17582	28/12/05	17:02	900
17583	28/12/05	17:03	100
17584	28/12/05	17:04	3000
17585	28/12/05	17:13	1000
17586	28/12/05	17:19	1500
17587	28/12/05	17:27	6200
17588	28/12/05	18:07	500
17589	28/12/05	18:09	1000
17590	28/12/05	18:10	500
17591	28/12/05	18:19	500
17592	28/12/05	18:22	2000
17593	28/12/05	18:30	9000
17594	28/12/05	18:31	1000
17595	28/12/05	18:32	900
17596	28/12/05	18:34	1200
17597	28/12/05	18:41	1500
17598	28/12/05	18:54	3500
17599	28/12/05	18:57	3000
17600	28/12/05	19:03	5500
17601	28/12/05	19:10	300
17602	28/12/05	19:17	1000
17603	28/12/05	19:36	6200
17604	28/12/05	19:37	2000

17605	28/12/05	19:38	3000
17606	28/12/05	19:57	10000
17607	28/12/05	20:15	500
17608	28/12/05	20:20	500
17609	28/12/05	20:33	20000
17610	28/12/05	20:37	2500
17611	28/12/05	20:50	2000
17612	28/12/05	21:05	500
17613	28/12/05	21:06	200
17614	28/12/05	21:07	500
17615	28/12/05	22:03	1000
17616	28/12/05	22:42	20000
17617	29/12/05	08:26	1000
17618	29/12/05	09:23	1000
17619	29/12/05	09:34	3000
17620	29/12/05	09:36	3000
17621	29/12/05	09:54	300
17622	29/12/05	09:56	4300
17623	29/12/05	10:32	2000
17624	29/12/05	10:49	2900
17625	29/12/05	10:51	3000
17626	29/12/05	11:01	100
17627	29/12/05	11:04	3000
17628	29/12/05	11:05	3000
17629	29/12/05	11:06	20000
17630	29/12/05	11:08	5000
17631	29/12/05	11:09	5000
17632	29/12/05	11:12	3000
17633	29/12/05	11:13	3000
17634	29/12/05	11:14	2000
17635	29/12/05	11:18	5000
17636	29/12/05	11:29	300
17637	29/12/05	11:32	8000
17638	29/12/05	11:34	15000
17639	29/12/05	11:46	6500
17640	29/12/05	11:49	200
17641	29/12/05	11:50	7000
17642	29/12/05	11:55	3000
17643	29/12/05	11:57	500
17644	29/12/05	11:57	500
17645	29/12/05	12:11	3000
17646	29/12/05	12:17	1500
17647	29/12/05	12:22	22000
17648	29/12/05	12:30	2000
17649	29/12/05	13:19	100
17650	29/12/05	13:21	2000
17651	29/12/05	13:39	200
17652	29/12/05	14:35	100
17653	29/12/05	14:43	5000
17654	29/12/05	14:47	7000
17655	29/12/05	14:49	200
17656	29/12/05	14:52	2000
17657	29/12/05	15:03	1000
17658	29/12/05	16:01	8000
17659	29/12/05	16:19	5000
17660	29/12/05	16:52	3000
17661	29/12/05	16:53	1000
17662	29/12/05	16:58	2000
17663	29/12/05	17:00	3000
17664	29/12/05	17:02	200
17665	29/12/05	17:10	1000
17666	29/12/05	17:11	5000
17667	29/12/05	17:12	2000
17668	29/12/05	17:14	5000
17669	29/12/05	17:18	500
17670	29/12/05	17:21	300
17671	29/12/05	17:26	200
17672	29/12/05	17:39	2000
17673	29/12/05	17:40	500
17674	29/12/05	17:42	5000
17675	29/12/05	17:58	2500
17676	29/12/05	18:09	500
17677	29/12/05	18:18	3000
17678	29/12/05	18:19	1000
17679	29/12/05	18:21	1000
17680	29/12/05	18:27	3000
17681	29/12/05	18:28	1000
17682	29/12/05	18:31	5000
17683	29/12/05	18:34	10000
17684	29/12/05	18:38	1000
17685	29/12/05	18:49	2000
17686	29/12/05	18:51	7200
17687	29/12/05	19:00	5000
17688	29/12/05	19:11	3000
17689	29/12/05	19:19	3500
17690	29/12/05	19:21	300
17691	29/12/05	19:25	2000
17692	29/12/05	19:33	1000
17693	29/12/05	19:35	5000
17694	29/12/05	19:38	500
17695	29/12/05	19:40	100
17696	29/12/05	19:44	10000
17697	29/12/05	20:13	5000
17698	29/12/05	20:14	2000
17699	29/12/05	20:31	3000
17700	29/12/05	20:36	1000

17701	29/12/05	23:17	100
17702	30/12/05	08:01	200
17703	30/12/05	08:37	10000
17704	30/12/05	09:36	8000
17705	30/12/05	09:39	200
17706	30/12/05	09:40	100
17707	30/12/05	09:52	5000
17708	30/12/05	09:54	2000
17709	30/12/05	09:58	3000
17710	30/12/05	10:19	20000
17711	30/12/05	10:20	8000
17712	30/12/05	10:24	500
17713	30/12/05	10:28	3000
17714	30/12/05	10:31	10000
17715	30/12/05	10:38	1500
17716	30/12/05	10:43	500
17717	30/12/05	10:49	6500
17718	30/12/05	11:04	1000
17719	30/12/05	11:05	500
17720	30/12/05	11:12	1000
17721	30/12/05	11:30	8000
17722	30/12/05	11:37	2000
17723	30/12/05	11:43	3000
17724	30/12/05	11:46	3000
17725	30/12/05	11:55	300
17726	30/12/05	12:38	100
17727	30/12/05	17:34	8000
17728	30/12/05	17:37	20000
17729	30/12/05	17:38	5000
17730	30/12/05	17:40	1000
17731	30/12/05	17:46	8000
17732	30/12/05	17:48	20000
17733	30/12/05	17:59	1000
17734	30/12/05	18:15	500
17735	30/12/05	18:16	300
17736	30/12/05	18:27	5000
17737	30/12/05	19:21	5000
17738	30/12/05	19:28	300
17739	30/12/05	19:31	3000
17740	30/12/05	19:58	500
17741	30/12/05	20:03	13000
17742	30/12/05	20:15	5000
17743	30/12/05	20:16	5000
17744	30/12/05	20:16	5000
17745	30/12/05	20:18	1500
17746	30/12/05	20:19	200
17747	30/12/05	20:21	5000
17748	30/12/05	20:22	5000
17749	30/12/05	20:24	200
17750	30/12/05	20:27	5000
17751	30/12/05	20:49	300
17752	30/12/05	21:02	1500
17753	30/12/05	21:18	200
17754	31/12/05	06:59	1900
17755	31/12/05	07:04	100
17756	31/12/05	07:43	2200
17757	31/12/05	10:08	10000
17758	31/12/05	10:10	2000
17759	31/12/05	10:11	2000
17760	31/12/05	10:15	2500
17761	31/12/05	10:22	500
17762	31/12/05	10:43	3000
17763	31/12/05	10:47	2000
17764	31/12/05	10:50	10000
17765	31/12/05	10:51	8000
17766	31/12/05	11:01	7000
17767	31/12/05	11:06	100
17768	31/12/05	11:09	2500
17769	31/12/05	11:18	500
17770	31/12/05	11:20	500
17771	31/12/05	11:22	100
17772	31/12/05	11:23	5000
17773	31/12/05	11:25	10000
17774	31/12/05	11:28	2000
17775	31/12/05	11:37	20000
17776	31/12/05	11:39	3000
17777	31/12/05	11:40	3000
17778	31/12/05	11:43	8000
17779	31/12/05	11:47	800
17780	31/12/05	11:49	10000
17781	31/12/05	11:50	300
17782	31/12/05	11:52	300
17783	31/12/05	11:59	9000
17784	31/12/05	12:01	5000
17785	31/12/05	12:10	1000
17786	31/12/05	12:11	500
17787	31/12/05	12:14	1000
17788	31/12/05	12:16	1000
17789	31/12/05	12:31	700
17790	31/12/05	12:34	3000
17791	31/12/05	12:46	5000
17792	31/12/05	12:58	5000
17793	31/12/05	12:59	3500
17794	31/12/05	13:00	3000
17795	31/12/05	13:02	800
17796	31/12/05	13:05	1000

17797	31/12/05	13:05	500	17811	31/12/05	17:43	5000
17798	31/12/05	13:14	3000	17812	31/12/05	17:44	2000
17799	31/12/05	13:28	500	17813	31/12/05	17:47	10000
17800	31/12/05	14:17	5000	17814	31/12/05	17:53	3000
17801	31/12/05	14:29	5000	17815	31/12/05	18:02	1000
17802	31/12/05	14:30	5000	17816	31/12/05	18:05	1000
17803	31/12/05	14:32	5000	17817	31/12/05	18:21	10000
17804	31/12/05	14:33	5000	17818	31/12/05	18:23	3000
17805	31/12/05	15:03	1000	17819	31/12/05	18:30	5000
17806	31/12/05	15:27	2500	17820	31/12/05	19:10	15000
17807	31/12/05	15:55	700	17821	31/12/05	20:14	10000
17808	31/12/05	17:20	1000	17822	31/12/05	20:38	500
17809	31/12/05	17:25	15000	17823	31/12/05	20:58	2000
17810	31/12/05	17:37	2000	17824	31/12/05	21:02	4100

Remark: As total number of hits is 17824, we did not show complete data because of space. If any one require we can provide it as we have stored as soft copy.